IMPLEMENTASI JARINGAN IPV6 DAN *CONSTRAINED APPLICATION PROTOCOL* (COAP) PADA SISTEM *MONITORING* KUALITAS AIR

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
M. WINGGA WOGGIASWORO
NIM: 125150307111015

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI JARINGAN IPV6 DAN *CONSTRAINED APPLICATION PROTOCOL* (COAP) PADA SISTEM *MONITORING* KUALITAS AIR

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
M. Wingga Woggiasworo
NIM: 125150307111015

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T., M.Eng.	Dahnial Syaugy, S.T., M.T., M.Sc.
NIP: 19820809 201212 1 004	NIK: 201607 870423 1 002

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

ii

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 19 Juli 2018

M. Wingga Woggiasworo

NIM: 125150307111015

iii

IDENTITAS PENGUJI

- Dosen Penguji I

  Rizal Maulana, S.T., M.T., M.Sc.

  NIK. 201607 891009 1 000

- Dosen Penguji II

  Adhitya Bhawiyuga, S.Kom., M.Sc.

  NIK. 201405 890720 1 001

DAFTAR RIWAYAT HIDUP

Nama	:	Muhammad Wingga Woggiasworo
Tempat, Tanggal Lahir	:	Sukoharjo, 8 Juli 1994
Alamat Asal	:	Puri Cendana, Jl. Taman Bromo 6 No. 9. Tambun Selatan, Kab. Bekasi. Jawa Barat.
Nama Orang Tua	:	Wagino
Riwayat Pendidikan	:	SD Negeri 4 Margajaya Kota Bekasi (2000-2002)
		SD Negeri 1 Mangunjaya Tambun Selatan, Kab. Bekasi (2002-2006)
		SMP Negeri 1 Tambun Selatan, Kab. Bekasi (2006-2009)
		SMK Negeri 2 Cikarang Barat, Kab. Bekasi (2009-2012)
		S1 Teknik Informatika Universitas Brawijaya (2012-2018)
Alamat di Malang	:	Jl. Ikan Paus VII No. 23 Blimbing, Kota Malang
No. Telpon/HP	:	081 232 533 589
E-mail	:	wm.wingga@gmail.com
Prestasi	:	-
Pengalaman Kepanitiaan	:	Lomba Karya Sistem Komputer (BASIK) II 2014
Pengalaman Organisasi	:	-

UCAPAN TERIMA KASIH

Ucapan terima kasih penulis haturkan kepada seluruh pihak yang selama ini telah mendukung dan membantu dalam proses penelitian skripsi. Kepada dosen pembimbing I dan dosen pembimbing II yang telah meluangkan waktunya dalam membimbing dan memberikan masukan berharga bagi skripsi ini. Ucapan terima kasih juga penulis haturkan kepada teman-teman di kampus yang telah memberikan dukungan dan berjuang bersama-sama dalam menyelesaikan penelitian. Semoga segala jerih payah perjuangan yang telah dilakukan bisa memberi manfaat bagi banyak orang.

Malang, 3 Agustus 2018

Penulis

wm.wingga@gmail.com

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah memberikan rahmat, taufik dan hidayah-Nya sehingga laporan penelitian dengan judul "Implementasi Jaringan IPv6 dan Constrained Application Protocol pada Sistem Monitoring Kualitas Air" ini dapat terselesaikan. Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan banyak pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Kedua orang tua yang saya cintai, yang telah sabar menunggu dan selalu memberikan do'a, motivasi, kasih sayang, dan dukung moril maupun materil sebagai penyemangat dalam menyelesaikan skripsi ini.
2. Bapak **Wayan Firdaus Mahmudy, S.Si., M.T,. Ph.D.** selaku Dekan Fakultas Ilmu Komputer
3. Bapak **Heru Nurwarsito, Ir., M.Kom.** selaku Wakil Dekan I Bidang Akademik Fakultas Ilmu Komputer.
4. Bapak **Tri Astoto Kurniawan, S.T., M.T., Ph.D.** selaku Ketua Jurusan Teknik Informatika.
5. Bapak **Sabriansyah Rizkiqa Akbar, S.T., M.Eng.** selaku Ketua Program Studi Teknik Komputer dan dosen pembimbing I yang dengan sabar membimbing dan mengarahkan penulis hingga dapat menyelesaikan skripsi ini.
6. Bapak **Dahnial Syauqy, S.T., M.T., M.Sc.** selaku dosen pembimbing II yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
7. Segenap Bapak Ibu dosen dan staf serta karyawan Fakultas Ilmu Komputer Universitas Brawijaya atas segenap ilmu pengetahuan yang diberikan.
8. **M. Kholis Fikri, Poby Zaarifwandono, Embris Nuresalandis, Ponco Wiguna, Arycca Septian Mulyana, M. Redi Fauzan, Dhiya N. Qurrotu'ain, Siti Aisyah, Dadan H. Ramdani, Ardy Frayogi, Aras Nizamul A. Anwar, Fauzi Awal Ramadhan, Jodie Putra Kahir, Samkhya Aparigraha, Iqbal Yuan Avisena, Siti Rizky Amalia, Novanda N., Rahmat Yanuar Putra, Galang Eiga Prambudi** terima kasih untuk dukungan dalam penyelesaian skripsi ini serta seluruh teman-teman sistem komputer angkatan 2012 dan pihak yang tidak bisa disebutkan satu per satu.
9. **Keluarga Besar TKJ1** dan teman-teman di Bekasi yang tidak bisa disebutkan satu per satu.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 19 Juli 2018

Penulis

wm.wingga@gmail.com

ABSTRAK

M. Wingga Woggiasworo, Implementasi Jaringan IPv6 dan *Constrained Application Protocol* (CoAP) Pada Sistem *Monitoring* Kualitas Air

Pembimbing: Sabriansyah Rizkiqa Akbar, S.T., M.Eng. dan Dahnial Syauqy, S.T., M.T., M.Sc.

Perkembangan IoT dalam penerapan protokol jaringan perangkat cerdas yang secara umum mengaktifkan penggunaan *internet protocols*. IPv6 akan mampu berkomunikasi dengan perangkat cerdas yang melekat karena ruang alamat yang cukup besar. Jaringan terbatas seperti 6LoWPAN terdiri dari perangkat yang sesuai dengan perangkat *IEEE 802.15.4* dan memiliki karakteristik untuk jaringan tidak stabil dengan *low bit-rate*, dan *low cost*. Pengaplikasian *sensor-node* berbasis 6LoWPAN menggunakan protokol CoAP sebagai media pertukaran pesan *web transfer protocol* untuk perangkat dan lingkungan yang terbatas (mis. *low-power* dan jaringan tidak stabil). Dengan berkaca pada permasalahan untuk kebutuhan *monitoring* kualitas air dalam berbagai industri, dilakukan implementasi *sensor-node* berbasis 6LoWPAN dan CoAP untuk sistem *monitoring* kualitas air dengan parameter sensor untuk mengetahui suhu air, kadar ph dan tingkat kekeruhan. Hasil akurasi dari penggunaan sensor-sensor tersebut yakni 95,9% untuk sensor suhu DS18B20. Rata-rata nilai akurasi 91,61% untuk sensor pH yang didapat melalui perbandingan dengan *bufferkit pH*. Dan sensor photodioda dengan rentang nilai 1%-20% untuk jernih dan rentang nilai 80%-100% untuk keruh yang digunakan sebagai pendeteksi tingkat kekeruhan. Dengan penerapan jaringan 6LoWPAN dan protokol CoAP dapat meminimalkan kesalahan pengiriman karena menyediakan *resource* protokol. Rata-rata *delay* dan *packet loss* yang diperoleh dari jaringan 6LoWPAN berdasarkan ukuran paket 56 *bytes* dan 128 *bytes* adalah 13.4 *ms* dan 20.1 *ms* dengan *packet loss rate* 0% dan 0% dalam jarak 5 meter. Untuk jarak terjauh yakni 40 meter, rata-rata *delay* adalah 30.8 *ms* dan 39.7 *ms* dengan *packet loss rate* 31% dan 57% berdasarkan ukuran paket 56 *bytes* dan 128 *bytes*. Sedangkan performa pengiriman pesan antar *node* berbasis 6LoWPAN dan CoAP mendapatkan waktu pengiriman 13,417 detik untuk jarak antar *node* yakni 45 meter dan 1,331 detik untuk jarak antar *node* yakni 5 meter.

Kata kunci: *WSN, 6LoWPAN, CoAP, MRF24J40MA, RaspberryPi*

ABSTRACT

M. Wingga Woggiasworo, ***Implementation of IPv6 Network and Constrained Application Protocol for Water Quality Monitoring System***

Mentor: Sabriansyah Rizkiqa Akbar, S.T., M.Eng. dan Dahnial Syauqy, S.T., M.T., M.Sc.

The development of IoT in the implementation of smart device network protocols that generally enable internet protocols. IPv6 will be able to communicate with smart devices that are attached because the address space is quite large. Limited networks such as 6LoWPAN consist of devices that are compatible with IEEE 802.15.4 devices and have characteristics for unstable networks with low bit-rates, and low costs. Application of sensor-node based on 6LoWPAN uses the CoAP protocol as a medium for web messages transfer protocol for limited devices and environments (eg: low-power and unstable networks). Reflecting on the problems for water quality monitoring needs in various industries, the implementation of sensor-nodes based on 6LoWPAN and CoAP for water quality monitoring systems with sensor parameters to determine water temperature, pH level and turbidity level. Accuracy results from the use of these sensors are 95.9% for DS18B20 which is a temperature sensor. The average accuracy of 91.61% for the pH sensor is obtained by comparison with pH bufferkit. And the photodiode sensor with a value range of 1% -20% for clear and 80% -100% value range for murky which is used to detect turbidity levels. With the implementation of the 6LoWPAN network and the CoAP protocol, it can minimize shipping errors because it provides protocol resources. Average delay and packet loss obtained from the 6LoWPAN network based on packet size of 56 bytes and 128 bytes is 13.4 ms and 20.1 ms with packet loss rate is 0% and 0% in distance of 5 meters. For the farthest distance of 40 meters, average delay is 30.8 ms and 39.7 ms with packet loss rate is 31%; 57% based on packet size 56 bytes and 128 bytes. While the message delivery performance between nodes based on 6LoWPAN and CoAP gets a delivery time of 13.417 seconds for the inter node distance of 45 meters and 1.331 seconds for the distance between nodes of 5 meters.

Keywords: *WSN, 6LoWPAN, CoAP, MRF24J40MA, RaspberryPi*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

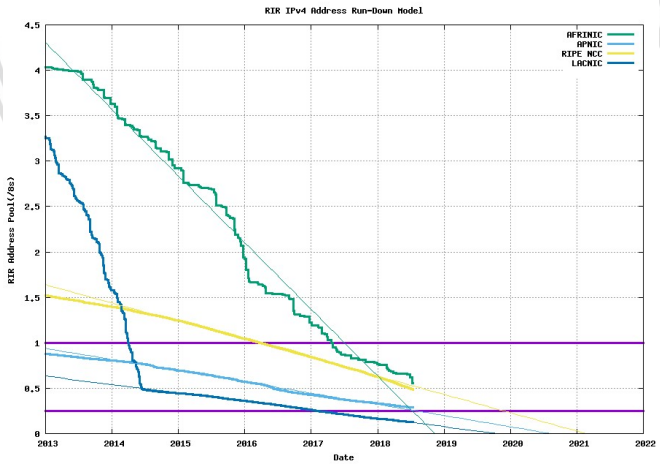
DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

Bab ini menguraikan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat, serta sistematika penulisan yang berkaitan pada penelitian ini sebagai tugas akhir kuliah (skripsi).

1.1 Latar Belakang

Internet of Things (IoT) melibatkan interaksi antar beragam perangkat yaitu perangkat sensor, agregator, aktuator dan *web* maupun *mobile application*. Implementasi IoT terdiri dari dua komponen utama yaitu *internet* dan *things*. *Internet* merupakan beragam infrastruktur jaringan yang digabungkan dalam skala masif dan berkembang dinamis berdasarkan standar dan protokol komunikasi. Sedangkan *things* merupakan benda atau perangkat yang memiliki identitas, atribut, karakteristik dan dapat berkomunikasi satu sama lain melalui sebuah *interface*. Terkait pengembangan dalam penerapan protokol jaringan, hal senada diungkapkan oleh (Shelby & Bormann, 2011) yang menyebutkan bahwa visi dibalik *internet of things* adalah *embedded devices* atau perangkat cerdas yang secara umum mengaktifkan penggunaan *internet protocols* dan merupakan bagian dari internet. Revolusi tersebut telah dimulai sejak 1990an pada sistem otomatisasi industri yang kemudian cepat digantikan oleh berbagai jenis *ethernet* industry. Dan menjadikan protokol internet lebih banyak digunakan antara perangkat cerdas tertanam dan sistem *back-end*. Tren ini akan berlanjut disemua segmen perangkat cerdas lainnya dengan penerapan *ethernet* dan *internet protocols* menjadi *ubiqutos* pada *things* dengan penerapan jaringan IPv6.



Gambar 1.1 RIR IPv4 Address Run-Down
sumber: (Potaroo, 2017)

Internet Protocol version 6 (IPv6) akan mampu berkomunikasi dengan perangkat yang melekat pada hampir semua benda buatan manusia karena ruang alamat yang sangat besar dari protokol IPv6. Sistem ini dapat membangun sebuah objek dalam skala yang besar. Hal tersebut didukung pula dengan Gambar 1.1 yakni alokasi penggunaan IPv4 berdasarkan zona wilayah yang terus mengalami

penurunan di setiap tahunnya. Oleh karena itu, peralihan yang sedang dilakukan saat ini dari IPv4 menuju IPv6 baik juga dilakukan pada perangkat cerdas tertanam. Dokumentasi dalam RFC 4919 (IETF, 2009) tentang jaringan terbatas seperti *IPv6 over Low-Power Wireless Personal Area Network* (6LoWPANs) terdiri dari perangkat yang sesuai dengan perangkat IEEE 802.15.4 yang memiliki karakteristik jarak jangkauan lebih kecil namun *low bit-rate*, *low power* dan *low cost*. Banyak perangkat yang menggunakan radio IEEE 802.15.4 akan membatasi dalam komputasi daya, memori dan *energy availability*.

Penelitian lain yaitu SOSG oleh (Schmidt, 2016) berpendapat bahwa 6LoWPAN sebagai sebuah *open IoT network protocol* yang kedepannya akan optimal dengan adanya *linux-wpan* yang dapat mendukung terhadap perangkat terdahulu maupun adanya pembaruan perangkat dan berpotensi terhadap industri. Beberapa penelitian tersebut telah dilakukan untuk merumuskan bagaimana pengaplikasian *Internet of Things* terhadap *embedded systems* agar dapat berhubungan antar perangkat yang bisa diimplementasikan dari beberapa *sensor-node* berbasis 6LoWPAN dengan menggunakan salah satu protokol komunikasi yang dipilih sebagai media pertukaran pesan. *Constrained Application Protocol* (CoAP) dinilai cukup optimal karena merupakan *web transfer protocol* untuk perangkat dan lingkungan yang terbatas (mis. *low-power* dan jaringan tidak stabil). Perangkat ini biasanya mempunyai 8-bit microcontroller dengan RAM dan ROM yang sangat kecil. Protokol ini didesain untuk komunikasi *Machine-to-machine* (M2M) seperti *smart-energy* dan *building automation*. Standar protokol CoAP didokumentasikan dalam RFC 7252 (IETF, 2014).

Dari pembahasan sebelumnya, maka penelitian diusulkan berupa *sensor-node* yang diimplementasikan *network* protokol 6LoWPAN dan protokol komunikasi CoAP yang berorientasi pada permasalahan pemantauan kualitas air. Air merupakan salah satu kebutuhan paling vital dalam berbagai proses metabolisme kehidupan dari makhluk hidup. Limbah domestik dan industri di kawasan kota atau dari kawasan kota juga dapat mempengaruhi kualitas air tersebut. Sebagai contoh adalah pada industri yang memang sangat membutuhkan data akurat kualitas air seperti PDAM, pertambakan, dan sebagainya. Meningkatnya pembangunan serta pesatnya pertumbuhan industri tentunya berdampak pada peningkatan beban limbah yang berdampak pada penurunan mutu air. Sehingga dilakukan implementasi sebuah sistem untuk memantau kualitas air ONLIMO oleh Badan Pengkajian dan Penerapan Teknologi (BPPT, 2012).

Oleh karena itu, kebutuhan sistem pemantauan kualitas air masih sangat dibutuhkan seiring dengan perkembangan teknologi. Dengan penerapan jaringan 6LoWPAN sebagai ruang lingkup *sensor-node* berbasis pengalamatan *Internet Protocol* (IP), mampu mengatasi permasalahan komunikasi dengan perangkat sensor IoT menggunakan protokol CoAP. Dan mendukung keakuratan perolehan data kualitas air yang dipublikasikan secara otomatis dengan beberapa parameter kualitas air yakni kadar ph air, kekeruhan air, dan suhu di dalam air.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang dibahas, terdapat beberapa rumusan masalah yang dapat dikaji adalah sebagai berikut:

1. Bagaimana mengimplementasikan jaringan IPv6 dengan protocol CoAP pada sistem *monitoring* kualitas air berdasarkan parameter nilai ph, kekeruhan, dan suhu?
2. Bagaimana menerapkan mekanisme komunikasi dengan protokol CoAP pada *sensor-node* berbasis IPv6 yang diusulkan?
3. Bagaimana kinerja dari implementasi jaringan IPv6 dan protokol CoAP terhadap sistem *monitoring* kualitas air yang diusulkan?

1.3 Tujuan

Tujuan yang ingin dicapai dari penelitian ini adalah sebagai berikut:

1. Mengembangkan jaringan terbatas 6LoWPAN terhadap jaringan *sensor-node* dengan meminimalkan kesalahan pengiriman
2. Menerapkan pola *request/response model* pada protokol CoAP untuk menjembatani pertukaran data antar *node*.
3. Mengetahui kinerja implementasi yang dapat menjembatanin antara perangkat sensor dengan *user* dari segi hasil pengiriman data monitoring.

1.4 Manfaat

Penelitian ini menghasilkan *sensor-node* dengan penerapan 6LoWPAN yang dapat menerima data dari protokol CoAP. Dalam hal perkembangan *Internet of Things*, penerapan ini diyakini memperoleh manfaat yaitu:

1. Menekan beban infrastruktur bagi industri yang menerapkan *embedded system* pada bidangnya.
2. Memudahkan dalam melakukan pengembangan untuk berfokus pada perangkat IoT yang membutuhkan heterogenitas sensor.
3. Mengoptimalkan protokol CoAP pada pengembangan perangkat IoT yang ditujukan untuk kebutuhan fungsional sistem.

Selain itu, penelitian ini juga dapat berguna sebagai dasar atau rujukan penelitian-penelitian sejenis karena masih banyak aspek dalam penelitian ini yang dapat ditingkatkan atau dikembangkan. Manfaat lainnya yaitu membantu penulis untuk lebih memahami perkembangan IoT dan komponen lain di dalamnya terutama topik jaringan sensor nirkabel.

1.5 Batasan masalah

Untuk lebih memfokuskan penelitian ini agar pembahasan penelitian tidak menyimpang dari apa yang telah dirumuskan, maka penelitian ini memiliki Batasan-batasan masalah sebagai berikut:

1. Implementasi 6LoWPAN pada sistem ini menggunakan dua *node* yang berfungsi sebagai *sensor node* dan *border router* menggunakan *raspberry pi* yang berjalan pada *kernel linux versi 4.7*.

2. Dalam penerapannya, faktor keamanan dari protokol CoAP yang digunakan belum diimplementasikan, karena hal tersebut bisa menjadi topik penelitian tersendiri nantinya dengan membandingkan protokol IoT yang lainnya.
3. Pembahasan difokuskan pada penerapan 6LoWPAN dan CoAP untuk dilakukan pengujian menggunakan jaringan *local* yang dikaji dari segi *integrating* sistem dan hasil monitoring data kualitas air.
4. Jangkauan untuk penggunaan *sensor-node* dan *node border router* berada pada satu lingkup area jaringan.

1.6 Sistematika pembahasan

Sistematika penulisan penelitian ditujukan untuk memberikan gambaran dan uraian dari penyusunan skripsi secara garis besar agar dapat membantu pembaca dalam memahami sistematika pembahasan isi dalam skripsi ini. Bagian ini berisi struktur penelitian skripsi mulai Bab Pendahuluan sampai Bab Penutup beserta deskripsi singkat dari masing-masing bab.

Bab I : Pendahuluan

Bab ini menguraikan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat, serta sistematika penulisan yang berkaitan pada penelitian ini sebagai tugas akhir kuliah (skripsi).

Bab II : Landasan Kepustakaan

Landasan kepustakaan berisi uraian dan pembahasan tentang teori, konsep, model, metode, atau sistem dari literatur ilmiah yang berkaitan dengan topik tugas akhir dan pertanyaan penelitian. Landasan kepustakaan digunakan sebagai dasar perancangan, implementasi, dan Analisa hasil penelitian ini.

Bab III : Metodologi

Pada bab metodologi penelitian, dijelaskan langkah kerja dalam implementasi jaringan IPv6 dan protokol CoAP pada sistem *monitoring* kualitas air. Langkah-langkah yang dijelaskan disini meliputi studi literatur, analisis kebutuhan, perancangan arsitektur sistem, implementasi, pengujian dan analisis serta pengambilan kesimpulan dan saran.

Bab IV : Rekayasa Kebutuhan

Pada bab ini membahas gambaran umum sistem yang menjelaskan kebutuhan fungsional dan fitur yang ada pada penerapan 6LoWPAN dan protokol CoAP pada sistem *monitoring kualitas air*. Selanjutnya menjelaskan rancangan arsitektur yang diterapkan agar perangkat sensor, *border router*, dan aplikasi saling berkomunikasi satu sama lain. Bab ini juga menjelaskan perancangan lingkungan penelitian yang berupa topologi jaringan serta metode dan skenario pengujian.

Bab V : Perancangan dan Implementasi

Bab ini menjelaskan tentang rancangan arsitektur yang diterapkan agar perangkat sensor, *border router*, dan aplikasi saling berkomunikasi satu sama lain dan perancangan lingkungan penelitian yang berupa topologi jaringan. Sedangkan proses implementasi rancangan arsitektur yang sudah dibuat menjadi *sensor-node* dan *node border router* pada mini komputer RaspberryPi dengan menampilkan *pseudocode* tentang proses-proses utama yang dibutuhkan oleh 6LoWPAN.

Bab VI : Pengujian dan Analisis

Tahapan pengujian dari 6LoWPAN dan protokol CoAP yang diimplementasikan pada sistem terbagi menjadi dua komponen pengujian yakni pada sistem integrasi dan sistem kualitas air. Sistem integrasi digunakan untuk menguji kebutuhan fungsional sistem. Pengujian ini dilakukan dengan menguji interaksi antar *node* yang telah dirancang untuk menjalankan satu fungsi tertentu. Sedangkan pengujian sistem kualitas air dilakukan untuk pengujian tingkat akurasi sensor yang digunakan dan telah dijabarkan pada bab perancangan. Kemudian dari hasil pengujian tersebut dilakukan analisis terhadap implementasi yang diharapkan.

Bab VII : Penutup

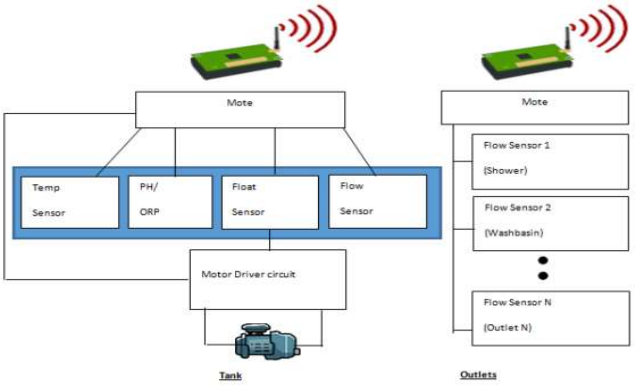
Pada bab ini terdapat kesimpulan yang diambil berdasarkan tahapan-tahapan yang sudah dilakukan mulai dari perancangan arsitektur, implementasi, pengujian serta analisa data. Kesimpulan akan menjawab pertanyaan-pertanyaan pada rumusan masalah. Bab ini juga menampung saran-saran untuk penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan berisi uraian dan pembahasan tentang teori, konsep, model, metode, atau sistem dari literatur ilmiah, yang berkaitan pada penelitian ini. Dilaksanakan sebagai tugas akhir kuliah (skripsi) yakni sistem *monitoring* kualitas air dengan mengimplementasikan *sensor-node* berbasis IPv6 dan *constrained application protocol* sebagai protokol komunikasi. Terdapat kajian pustaka yang menjelaskan secara umum penelitian-penelitian terdahulu yang berhubungan dengan topik penelitian. Menguraikan persamaan dan perbedaan terhadap penelitian terdahulu yang dibutuhkan sesuai dengan karakteristik penelitiannya dan syarat kecukupan keminatan tertentu.

2.1 Tinjauan Pustaka

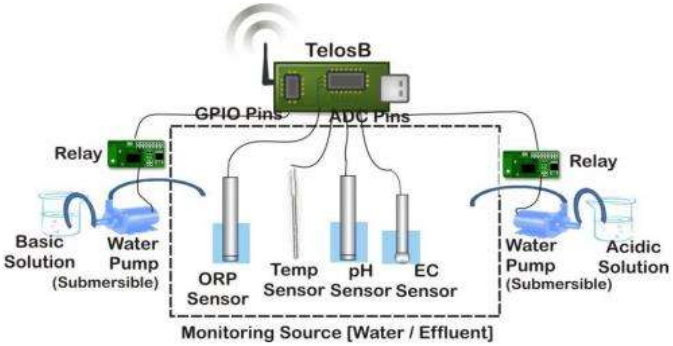
Terkait dengan perancangan dan implementasi *IPv6* dan *Constrained Application Protocol* pada sistem monitoring kualitas air, terdapat beberapa penelitian yang dijadikan sebagai rujukan awal penelitian. Berikut ini akan dijelaskan beberapa landasan dan referensi yang diperoleh dari penelitian-penelitian tersebut. Penelitian yang pertama yaitu *Design of IPv6 Network Enabled Smart Water Flow Meter System for India*. Dengan menggunakan Mote CC2538 Dev. Kits (Texas Instruments, 2014) yang dapat langsung diintegrasikan dengan 6LoWPAN sebagai pendukung jaringan IPv6, penelitian tersebut menerapkan pada sistem *water flow meter* untuk pemantauan kualitas air dan langsung memberikan tindakan kepada *water pump* sebagai *actuator* ketika kualitas air tidak memenuhi standar yang diterapkan peneliti. Namun, pada penelitian tersebut dinilai terlalu menggunakan perangkat yang sangat membebankan *cost production* sehingga memacu kepada pengembangan selanjutnya yang lebih dapat mengoptimalkan penerapan *tiny devices* (Ankith, et al., 2015).



Gambar 2.1 Logical Block of Transmitting Unit
sumber: (Ankith, et al., 2015)

Penelitian selanjutnya yang dijadikan rujukan adalah penelitian yang sebelumnya telah dijadikan acuan oleh (Ankith, et al., 2015) yakni *Design of 6LoWPAN enabled Real Time Water Quality Monitoring System using CoAP*. Perbedaan dengan penelitian sebelumnya, penelitian ini diterapkan dalam bidang

pertanian yang berhubungan pada kualitas air (*aquatic life*) untuk mencegah dari polusi terhadap air yang memiliki dampak berkepanjangan.



Gambar 2.2 Block Representation of WQMS Transmitter and Actuator
sumber: (Adarsh & Divya, 2014)

Perangkat yang digunakan menggunakan TelosB Mote karena membutuhkan perangkat yang banyak untuk tersebar di beberapa titik-titik pemantauan. Penelitian tersebut belum melakukan pengolahan data yang dihasilkan, sehingga data yang diakses oleh pengguna masih berupa *raw data* (data mentah) melalui *browser plugin californium* CoAP untuk menampilkan data yang melalui protokol CoAP (Adarsh & Divya, 2014).

Penelitian ketiga yang menjelaskan tentang analisis kinerja jaringan 6LoWPAN dengan menggunakan 1 *edge node* dan 1 *end node* terhadapat *protocol stack* 6LoWPAN *network*. Analisis yang dilakukan pada penelitiannya *Tunneling 6LoWPAN Protocol Stack in IPv6 Network* berdasarkan parameter pengujian *quality of service* yakni parameter *throughput, delay* dan *packet loss*. Penelitian tersebut menjadi usaha yang signifikan dalam evaluasi kinerja 6LoWPAN *protocol stack* pada jaringan IPv6 karena menerapkan faktor jarak sebagai pembanding terhadap parameter-parameter penilai dari *quality of service* (Alfadoni, 2016).

Berdasarkan penelitian yang telah dilakukan, menurut pandangan penulis bahwa potensi *wireless sensor network* dapat mangaplikasikan fungsi penting yang diintegrasikan dalam pengaplikasian Internet of Things. Oleh karena itu, penulis mengambil andil untuk turut melakukan pengembangan dalam implementasi *tiny devices sensor-node* menggunakan jaringan IPv6 dan protokol CoAP. Dalam hal ini, penulis melakukan penelitian dengan menggunakan *module wireless* yang berbeda dengan penelitian sebelumnya dan tidak menggunakan Mote atau Development Kit yang sebelumnya telah digunakan, melainkan menggunakan *mini* komputer Raspberry PI sebagai *edge node* dan *end node*. Dengan menggunakan beberapa kelebihan-kekurangan dalam penelitian sebelumnya, diharapkan dapat membuat yang lebih baik secara sistem dan pesan yang dihasilkan untuk pengguna.

2.2 Dasar Teori

Dasar teori berisi tentang teori-teori pendukung yang berhubungan dalam perancangan dan implementasi pada penelitian dengan topik sistem monitoring kualitas air menggunakan *Constrained Application Protocol* dan IPv6.

2.2.1 Internet Of Things

Internet of Things (IoT) diperkenalkan pertama sebagai wacana yang dipresentasikan dalam Procter & Gamble (P&G) pada tahun 1999 (Ashton, 2009). IoT dapat dijelaskan sebagai satu *set things* yang saling terkoneksi melalui internet. *Things* disini dapat berupa *tags*, sensor, manusia dll. IoT berfungsi mengumpulkan data dan informasi dari lingkungan fisik (*environment*), data-data ini kemudian akan diproses agar dapat dipahami maknanya.

Kemampuan dari IoT untuk saling berkomunikasi ini membuat IoT dapat diterapkan di segala bidang. Di bidang kesehatan (Lopez, 2013), sensor IoT dapat digunakan untuk memonitor kondisi pasien, sehingga kondisi pasien tetap terpantau selama 24 jam. Di bidang pertanian, IoT dapat digunakan sebagai sensor untuk memonitor kondisi tanah, suhu dan kelembapan yang penting bagi tanaman. Di bidang smart building, IoT dapat digunakan untuk memonitor penggunaan listrik tiap gedung (Chen, 2011). Selain itu IoT juga dapat digunakan di bidang automation, transportasi, smart grid dan lainnya.

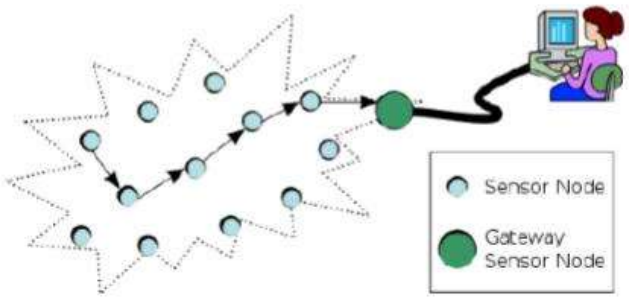
Menurut (Tan, 2014), teknologi dalam IoT dibagi menjadi beberapa arsitektur layer. Layer pertama yaitu Layer Perception, layer ini berfungsi membaca dan mengumpulkan informasi dari lingkungan fisik (*environment*). Kemudian, data akan dikirim ke *layer network*. Yang akhirnya data akan digunakan di dalam layer aplikasi. Perception Layer bertanggung jawab untuk mengkonversi data menjadi sinyal yang dikirim melalui network agar dapat dibaca oleh layer aplikasi. Sebagai contoh, penggunaan *barcode* oleh minimarket. Di dalam *barcode* tersebut terdapat data seperti nama, harga dan stok barang. Ketika informasi telah didapatkan, maka layer network akan bertanggung jawab untuk pengiriman data dari satu host ke host yang lain. Ada berbagai macam teknik yang digunakan seperti ZigBee, Wifi, *6lowpan* dll. Sedangkan layer aplikasi berfungsi untuk memproses informasi yang telah didapatkan untuk digunakan sesuai keperluannya.

2.2.2 6 over Low Power Wireless Personal Area Network (6LoWPAN)

Langkah pertama yang umum menuju IoT adalah mengubah jaringan pada protokol *proprietary* ke jaringan berbasis IP. Peningkatan besar pada ruang alamat merupakan faktor penting dalam pengembangan Internet of Things. Menurut (Steve Leibson, 2009), yang mengidentifikasi dirinya sebagai pemandu sesekali di Museum Sejarah Komputer, perluasan ruang alamat berarti kita bisa memberikan alamat IPV6 ke setiap atom di permukaan bumi, dan masih memiliki cukup alamat yang tersisa untuk melakukan 100+ bumi.

Dengan kata lain, manusia dapat dengan mudah menetapkan alamat IP untuk segala hal di planet ini. Peningkatan jumlah node cerdas, serta jumlah data hulu yang dihasilkan node, diharapkan dapat meningkatkan kekhawatiran baru tentang privasi data, kedaulatan data dan keamanan. Inovasi baru dalam teknologi protokol internet, yang disebut *6lowpan*, membuat *Internet of Things* menjadi hal yang kenyataan. 6LoWPAN adalah standar dari IETF, yang pertama kali diterbitkan pada tahun 2007 (Kushalnagar, 2007), yang mengoptimalkan *ipv6* untuk digunakan dengan teknologi komunikasi berdaya rendah dan berkapasitas rendah seperti IEEE 802.15.4.

6LoWPAN bekerja dengan menekan 60 *byte header* menjadi hanya 7 *byte*, dan mengoptimalkan mekanisme untuk jaringan nirkabel. Sensinode menyediakan solusi *stack* dan *router* 6LoWPAN untuk berbagai macam IEEE 802.15.4 dan teknologi radio berdaya rendah lainnya bersama dengan perutean *mesh* berbasis standar IETF. Perangkat berbasis IP dapat dihubungkan dengan mudah ke jaringan IP lain tanpa memerlukan *gateway* terjemahan atau *proxy*. Jaringan IP memungkinkan penggunaan infrastruktur jaringan yang ada. Teknologi berbasis IP telah ada selama beberapa dekade, sangat terkenal, dan telah terbukti bekerja dan berskala.



Gambar 2.3 Persebaran Node dan Akses Pada WSN
sumber: (trusolo.co)

Ada sejumlah besar aplikasi yang bisa mendapatkan keuntungan dari pendekatan Wireless Embedded Internet. Saat ini aplikasi ini diimplementasikan dengan menggunakan berbagai teknologi *proprietary* yang sulit untuk diintegrasikan ke dalam jaringan yang lebih besar dan dengan layanan berbasis Internet. Manfaat menggunakan protokol Internet dalam aplikasi ini dan dengan demikian mengintegrasikannya dengan *internet of things* termasuk (RFC4919, 2007):

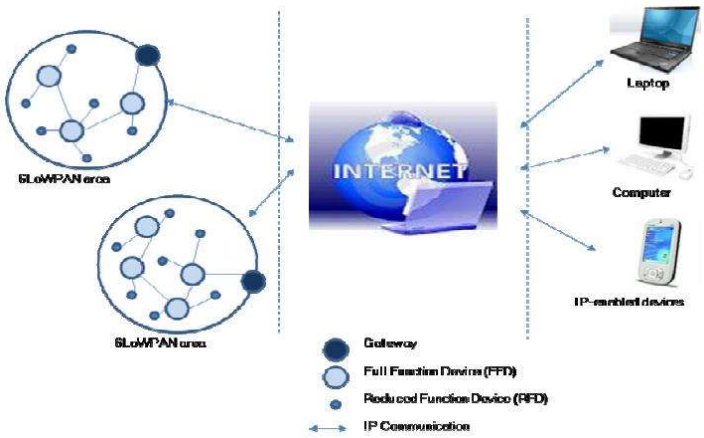
1. Perangkat berbasis IP dapat dihubungkan dengan mudah ke jaringan IP lain tanpa memerlukan *gateway* terjemahan atau *proxy*.
2. Jaringan IP memungkinkan penggunaan infrastruktur jaringan yang ada.
3. Teknologi berbasis IP telah ada selama beberapa dekade, sangat terkenal, dan telah terbukti bekerja dan berskala. API *socket* (*application programming interface*) adalah salah satu API yang paling terkenal dan banyak digunakan di dunia.

4. Teknologi IP ditentukan secara terbuka dan bebas, dengan standar proses dan dokumen tersedia bagi siapa saja. Hasilnya adalah teknologi IP mendorong inovasi dan lebih dipahami oleh khalayak yang lebih luas.
5. Alat untuk mengelola, commissioning dan mendiagnosis jaringan berbasis IP sudah ada.

2.2.2.2 Arsitektur 6LoWPAN

Protokol *stack 6lowpan* terdiri dari lapisan 802.15.4 *physical* (PHY) dan *medium access control* (MAC), lapisan adaptasi, lapisan jaringan, lapisan *transport* dan lapisan aplikasi. Kini, banyak penelitian tentang lapisan ini telah dibuat dan dianalisis. 802.15.4 PHY dan MAC sama persis dengan standar protokol ZigBee. Dalam penelitiannya (Nurul Halimatul Asmak Ismail, 2012), meninjau protokol *stack* di *6lowpan* diidentifikasi dan beberapa protokol routing dilibatkan. *6lowpan* adalah kependekan dari IPv6 melalui Low Power Wireless Area Networks. Spesifikasi *6lowpan* melibatkan transmisi jaringan IPv6 melalui IEEE 802.15.4.

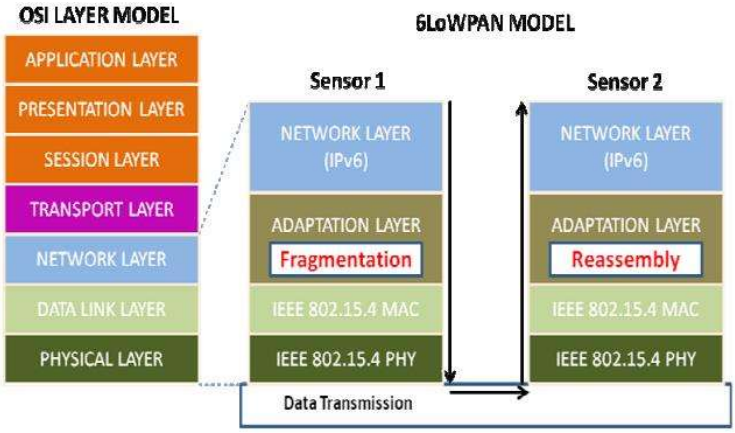
Ide awalnya adalah bahwa Protokol Internet dapat diterapkan pada perangkat kecil dan ringan. Jaringan area pribadi nirkabel (WPAN) adalah jenis jaringan nirkabel. Ini adalah jaringan untuk perangkat interkoneksi di sekitar area kerja tertentu dimana koneksi nirkabel. Skema pengalamatan lapisan jaringan dapat dikategorikan menjadi, revisi keempat dan keenam dalam pengembangan Protokol Internet (IP), IPv4 dan IPv6. Karena protokol IPv6 futuristik diciptakan untuk menggantikan dan mengganti IPv4, bersamaan dengan revolusi jaringan heterogen dan internet, *6lowpan* mulai hidup dalam menghindari penciptaan protokol 4LoWPAN yang belum lahir.



Gambar 2.4 6LoWPAN Overview
sumber: (Shelby & Bormann, 2011)

Perbedaan Model Lapisan OSI dan model referensi tumpukan protokol *6lowpan*. Ini mengadopsi lapisan standar PHY dan MAC IEEE 802.15.4 yang ditentukan pada penelitiannya (Kushalnagar, 2007). Model *6lowpan* terdiri dari lapisan aplikasi, lapisan presentasi, lapisan sesi, lapisan *transport*, lapisan jaringan, lapisan MAC IEEE 802.15.4 dan lapisan IEEE 802.15.4 PHY. Makalah ini akan membahas semua lapisan di *6lowpan*. Namun, ide utamanya adalah penekanan

lapisan jaringan dan adaptasi karena merupakan kerangka utama dari *6lowpan*. Lapisan *stack 6lowpan* terdiri dari lapisan PHY, lapisan MAC, lapisan adaptasi, lapisan jaringan, lapisan transport dan lapisan aplikasi. Pada dasarnya, ia menggunakan bagaimana perangkat IEEE 802.15.4 berkomunikasi satu sama lain melalui saluran nirkabel. Perbedaan utama antara protokol *stack 6lowpan* dan OSI adalah munculnya lapisan adaptasi. Lapisan ini diasumsikan melakukan fragmentasi atau *reassembly*, kompresi header dan pengalamatan mesh.

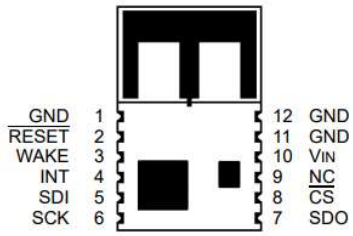


Gambar 2.5 6LoWPAN Layer Model dan OSI Layer Model
sumber: (Kushalnagar, 2007)

Munculnya *6lowpan* karena adanya jaringan heterogenitas perangkat (*things*), integrasi antara protokol lapisan jaringan yang lebih tinggi dari IPv6 dan protokol lapisan MAC yang lebih rendah dari IEEE 802.15.4. Oleh karena itu, ia menerapkan beberapa modifikasi pada tumpukan protokol yang ada dan mengenalkan *stack* protokol *6lowpan*. Selain itu, beberapa perbandingan antara IPv4 dan IPv6 telah dibuat dan ditinjau ulang. Selain itu, dua skema *routing* diidentifikasi: Route-over dan Mesh-under serta skema ini diterapkan pada lapisan adaptasi *6lowpan* dan lapisan jaringan.

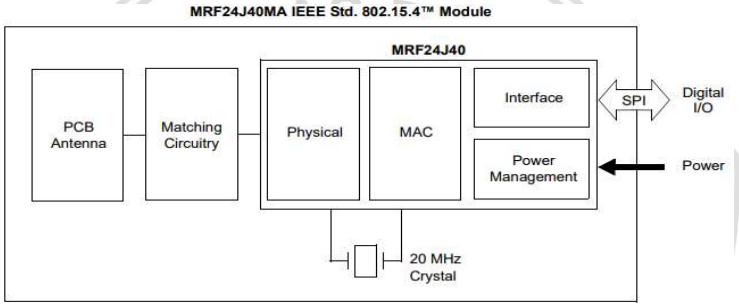
2.2.3 Wireless Module MRF24J40MA

MRF24J40MA adalah IEEE Std 2.4 GHz. 802.15.4. *compliant, surface mount module dengan integrated crystal, regulator tegangan internal, matching circuitry dan PCB antenna.* Modul MRF24J40MA beroperasi pada pita frekuensi 2,4 GHz yang tidak berlisensi dan kompatibel dengan FCC, IC dan ETSI. Desain modul terpadu membebaskan integrator dari desain RF dan antena yang luas, dan pengujian kepatuhan terhadap peraturan, yang memungkinkan waktu lebih cepat ke pasar. Modul MRF24J40MA kompatibel dengan tumpukan perangkat lunak Microchip ZigBee®, MiWi ™ dan MiWi P2P. Setiap tumpukan perangkat lunak tersedia sebagai unduhan gratis, termasuk kode sumber, dari situs microchip.



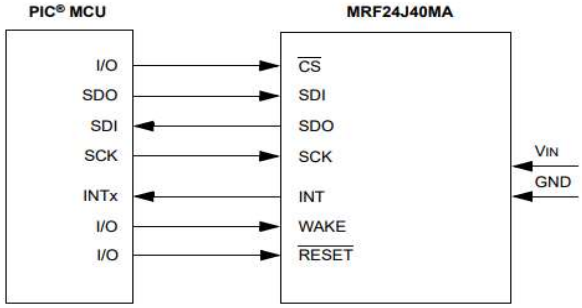
Gambar 2.6 Diagram Pin MRF24J40MA
sumber: (Microchip, 2008)

Modul MRF24J40MA telah menerima persetujuan pengaturan untuk perangkat modular di Amerika Serikat (FCC), Kanada (IC) dan Eropa (ETSI). Persetujuan modular menghilangkan kebutuhan akan antena RF dan antena yang mahal dan memungkinkan pengguna akhir memasang modul MRF24J40MA di dalam produk jadi dan tidak memerlukan pengujian peraturan untuk radiator(pemancar rf) yang disengaja (Microchip, 2008).



Gambar 2.7 Diagram Blok MRF24J40MA
sumber: (Microchip, 2008)

Diagram blok yang disederhanakan dari modul MRF24J40MA. Modul ini didasarkan pada teknologi microchip MRF24J40 IEEE 802.15. 2.4 GHz RF Transceiver IC. Antarmuka modul untuk mikrokontroler Microchip PIC yang populer melalui antarmuka SPI 4-kawat serial, mengganggu, bangun, Reset, power dan ground. Komunikasi data dengan modul MRF24J40MA didokumentasikan di Lembar Data Transisi RF "MRF24J40 IEEE 802.15.4 ™ 2.4 GHz" (Microchip, 2010). Lihat Lembar Data MRF24J40 untuk protokol antarmuka serial tertentu dan daftarkan definisi.



Gambar 2.8 MRF24J40MA Interface
sumber: (Microchip, 2010)

MRF24J40MA adalah IEEE STd 2.4 GHz yang lengkap. Modul *mount* permukaan 802.15.4 yang sesuai dengan kristal terintegrasi, regulator *voltase* internal, sirkuit pencocokan dan antena PCB. Antarmuka modul MRF24J40MA ke mikrokontroler Microchip PIC yang populer melalui antarmuka SPI 4-kawat serial, mengganggu, bangun, menyetel ulang, memberi daya dan ground. Komunikasi data dengan modul MRF24J40MA didokumentasikan di Lembar Data Transisi RF "MRF24J40 IEEE 802.15.4 2.4 GHz" (Microchip, 2010). Lihat Lembar Data MRF24J40 untuk protokol antarmuka serial tertentu dan daftarkan definisi.

Modul MRF24J40MA didasarkan pada Microchip Technology MRF24J40 IEEE 802.15.4 ™ 2.4 GHz RF Transceiver IC. Serial I / O (SCK, SDI, SDO dan CS), pin RESET, WAKE dan INT dibawa ke pin modul. Sinyal SDO adalah tri-state yang disangga oleh IC2 untuk memecahkan errata silikon, di mana sinyal SDO tidak dilepaskan ke keadaan impedansi tinggi, setelah pin CS kembali ke keadaan tidak aktif. Kristal, X1, adalah kristal 20 MHz dengan toleransi frekuensi ± 10 ppm @ 25 ° C untuk memenuhi IEEE Std. Toleransi simbol tingkat 802.15.4 ± 40 ppm. Balun dibentuk oleh komponen: L1, L3, C2 dan C14. L2 adalah RF tersedak dan pull-up untuk pin RFP dan RFN pada MRF24J40. C15 adalah kapasitor blok DC. Saringan low-pass dibentuk oleh komponen: L4, C16 dan C17. Kapasitor yang tersisa memberikan RF dan bypass digital.

2.2.4 Protokol Internet Of Things

Salah satu komponen utama yang menjadikan IoT dapat berfungsi dengan baik adalah protokol. Beberapa penelitian telah dilakukan untuk merumuskan dan melakukan standardisasi protokol untuk IoT. dari penelitian tersebut lahirnya protokol yang dikenal sekarang seperti CoAP, MQTT, BLE, WSN, RFID, dan seterusnya. Pada dasarnya, protokol-protokol tersebut terbagi menjadi dua kategori yakni protokol jaringan seperti IEEE 802.11, BLE, WSN, dan RFID. Kedua, protokol aplikasi seperti CoAP, MQTT, XMPP dan AMQ.

Penelitian yang dilakukan oleh (Al-Fuqaha et al, 2015), (Tuwanut, 2015) dan juga referensi dari (Sovani, 2017) tentang protokol pada IoT menyebutkan bahwa setidaknya terdapat tiga protokol utama yang digunakan dalam IoT. Protokol pertama yakni CoAP karena dapat digunkan pada perangkat dengan resource yang sangat terbatas. Selain itu protokol ini juga menerapkan semantik yang sama dengan HTTP. Protokol kedua yakni MQTT, protokol ini sangat tepat digunakan untuk komunikasi Machine-to-Machine atau untuk keperluan *real-time*. Protokol ini menerapkan pola *publish/subscribe* dalam implementasinya. Protokol ketiga yakni Websocket dimana protokol ini sangat tepat ketika digunakan untuk menghubungkan perngkat IoT seperti *middleware* dengan *internet* atau *cloud*.

Constrained Application Protocol (CoAP) merupakan protocol pada layer aplikasi untuk IoT yang dikembangkan oleh The IETF Constrained RESTful Environments (CoRE). CoAP dibangun berdasarkan pertukaran pesan pada web dengan semantik REpresentational State Transfer (REST) pada protokol HTTP. Untuk berkomunikasi, CoAP menggunakan kode pesan yang sama dengan HTTP seperti GET, POST, PUT dan DELETE. Hal yang membedakan keduanya adalah

dibandingkan dengan HTTP yang bekerja dengan TCP, CoAP bekerja pada UDP yang membuat protokol ini sangat pas untuk kebutuhan IoT. Lebih dalam lagi, pada CoAP beberapa fungsi pada HTTP diubah untuk kebutuhan IoT seperti ukuran header tiap pesan yang lebih kecil.

Protokol MQTT bertujuan untuk menghubungkan embedded devices seperti sensor network dengan aplikasi dan middleware. Koneksi MQTT menggunakan mekanisme routing (one-to-one, one-to-many dan many-to-many), hal ini membuat MQTT menjadi protokol yang optimal untuk IoT dan Machine to Machine (M2M). MQTT menggunakan pola publish/subscribe untuk menyediakan fleksibilitas dan kemudahan dalam implementasinya. MQTT dibangun di atas TCP dan memiliki dua jenis spesifikasi yakni MQTT v3.1 dan MQTT-SN.

Protokol websocket dan websocket API dihadirkan bersamaan dengan HTML 5 sebagai *upgrade* dari protokol HTTP untuk komunikasi data secara *real-time*. websocket menjadi satu-satunya *framework* pada web yang mendukung komunikasi secara *asynchronous* dan *full-duplex*. Terdapat banyak riset yang sudah dilakukan tenang perbandingan performa *websocket* dengan teknologi sejenis dan hasilnya adalah semua riset tersebut menyimpulkan bahwa *websocket* merupakan terobosan baru dalam protokol komunikasi secara *real-time* (Skvorc, et al., 2014). Sampai sekarang *websocket* sudah didukung oleh browser seperti Chrome, Safari, Opera, Firefox dan IE10 (Zhang & Shen, 2013).

Bluethooth Low Energy (BLE) merupakan salah satu protokol infrastruktur yang dikhususkan untuk keperluan IoT. Kelebihan BLE yaitu penggunaan daya yang lebih kecil, *latency* yang lebih rendah dan jarak jangkauan lebih jauh dibandingkan dengan Bluethooth biasa (Bluetooth Classic). Di sisi lain, kekurangan dari BLE adalah jumlah perangkat yang sudah kompatibel masih terbatas. Kebanyakan perangkat yang sudah kompatibel adalah *smartphone* terbaru atau perangkat khusus yang dalam penggunaannya membutuhkan BLE. Selebihnya, perangkat IoT saat ini masih bergantung pada teknologi Wifi sebagai protokol infrastruktur utama.

2.2.5 Constrained Application Protocol (CoAP)

Constrained Application Protocol (CoAP) merupakan *web transfer protocol* untuk perangkat dan lingkungan yang terbatas (mis. low-power dan jaringan tidak stabil). Perangkat ini biasanya mempunyai 8-bit *microcontroller* dengan RAM dan ROM yang sangat kecil, sedangkan jaringan terbatas seperti IPv6 over Low-Power Wireless Personal Area Networks (*6lowpan*s) sering kali memiliki tingkat kesalahan pengiriman yang tinggi dengan throughput hanya 10 kbps. Protokol ini didesain untuk komunikasi Machine-to-machine (M2M) seperti *smart-energy* dan *building automation*. Standar protokol CoAP didokumentasikan dalam (Shelby, et al., 2014) (RFC7252, 2016).

CoAP menggunakan *request/response* model untuk berinteraksi antar endpoint, mendukung fitur *discovery of services*, dan juga beberapa konsep utama dari web seperti URI dan tipe media. Salah satu tujuan dari CoAP adalah menyediakan protokol web untuk lingkungan yang terbatas, khususnya pada

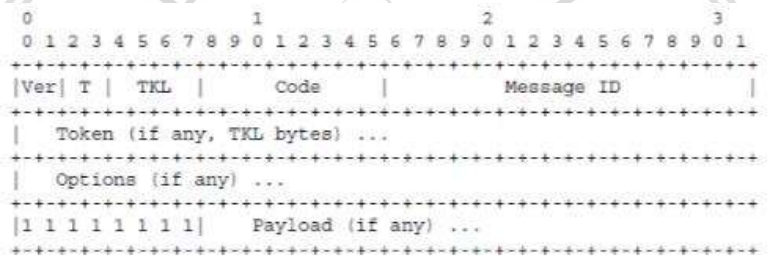
bidang energi, *building automation* dan komunikasi Machine-to-machine (M2M). CoAP bukan sekedar menyederhanakan protokol HTTP, melainkan mengambil bagian REST dari HTTP dan mengoptimasikannya untuk komunikasi M2M. Meskipun CoAP merupakan mimik dari HTTP, akan tetapi protokol ini memiliki fitur-fitur penting seperti built-in discovery, multicast dan pertukaran pesan secara asinkron.

Selain yang disebutkan sebelumnya, CoAP memiliki fitur penting lain yaitu:

1. Protokol web yang sesuai kebutuhan M2M dalam lingkungan yang terbatas.
2. Menggunakan UDP dengan pilihan reliability yang mendukung unicast dan multicast.
3. Pertukaran pesan secara asinkron serta overhead yang sangat rendah.
4. Mendukung URI, Content-Type, proxy dan cache layaknya HTTP.
5. Koneksi aman menggunakan Datagram Transport Layer Security (DTLS).

2.2.5.2 Messaging Model

Pada dasarnya CoAP dibuat berdasarkan pertukaran pesan antar endpoint menggunakan UDP, namun CoAP juga bisa digunakan pada transport protocol yang lain seperti Datagram Transport Layer Security (DTLS), SMS, TCP atau SCTP.



Gambar 2.9 CoAP Messaging Model
sumber: (IETF, 2014)

Sebagian mengira bahwa Perpesanan pada CoAP menggunakan pendekatan dua layer, pertama yaitu layer yang digunakan untuk koneksi UDP dan interaksi asinkron di dalamnya. Kedua yaitu layer untuk interaksi request/response menggunakan metode dan kode respon. Padahal sebenarnya CoAP merupakan satu protokol dengan *resquest/respond* sebagai fitur pada CoAP header.

Pesan dalam CoAP di-encode dalam sebuah format biner. Pesan ini diawali dengan 4 byte header. Bagian selanjutnya yaitu Token berukuran 0 – 8 byte untuk memastikan bahwa antara pesan yang dikirim dan diterima sesuai. Informasi terkait request/response seperti URL dan tipe media ada pada bagian selanjutnya yaitu Options. Terakhir adalah payload yang dikirimkan.

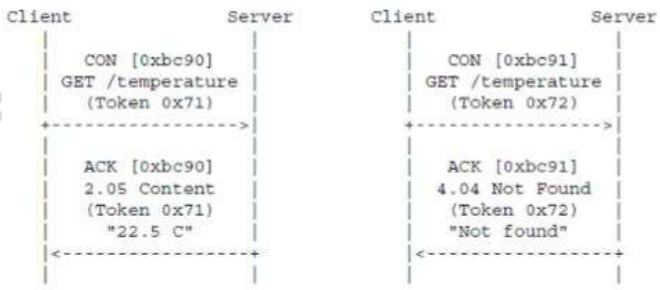
Tiap pesan mengandung Message ID (16 bit) untuk mendeteksi duplikasi apabila menggunakan opsi reliability. *Reliability* didapatkan dengan memberikan tanda Confirmable (CON) pada pesan yang dikirim. Dengan ini pengirim akan menunggu pesan Acknowledgement (ACK) dari penerima. Apabila dalam waktu tertentu pengirim tidak menerima pesan ACK maka pesan tersebut akan dikirimkan kembali. Sebaliknya, terjadi kesalahan pada penerima, misalkan

penerima tidak bisa memproses pesan yang diterima maka ia akan mengirim pesan Reset (RST).

Sebuah pesan yang tidak membutuhkan reliability dapat dikirimkan sebagai pesan Non-confirmable (NON). Pesan ini mengindikasikan bahwa penerima tidak harus mengirimkan pesan ACK pada pengirim, akan tetapi masih memungkinkan untuk mengirim pesan RST.

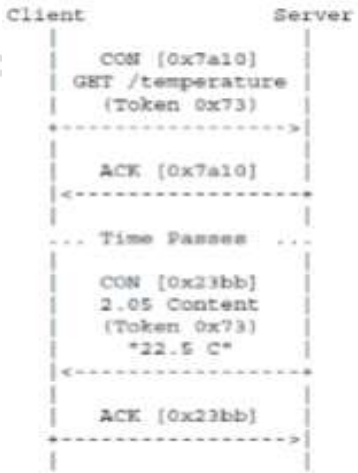
2.2.5.3 Request/Response Model

Sebuah pesan request baik itu CON maupun NON, jika memungkinkan, response akan diberikan bersamaan dengan pesan ACK. Kondisi ini dinamakan piggybacked response, apabila piggybacked response tidak sampai pada tujuan maka client akan mengirimkan request kembali. Contoh piggybacked response dapat dilihat pada gambar di bawah ini, satu berhasil dan lainnya mengembalikan 404 response.



Gambar 2.10 CoAP Request Response Skematik
sumber: (Anwari, 2016)

Apabila server tak dapat merespon CON request secara langsung, server mengirimkan pesan ACK sehingga client akan berhenti mengirimkan request. Apabila respon sudah siap maka server mengirimkan pesan CON pada client dan client akan membalas dengan pesan ACK. Model seperti disebut dengan Separate response dan dapat dilihat pada gambar di bawah ini.



Gambar 2.11 Paket Confirmable (CON)
sumber: (Anwari, 2016)

Jika request yang dikirim adalah Non-Confirmable (NON) maka respon yang digunakan adalah pesan NON juga, meskipun diperbolehkan juga server mengirimkan pesan CON.

Gambar 2.12 Paket Non-confirmable (NON)
sumber: (Anwari, 2016)

CoAP menggunakan method GET, POST, PUT dan DELETE sama seperti pada protokol HTTP untuk mengirimkan request. Pembahasan tentang method pada CoAP akan dijelaskan pada bagian selanjutnya (Anwari, 2016).

2.2.5.4 Definisi Method

Seperti yang sudah disebutkan sebelumnya bahwa CoAP menggunakan method yang sama dengan protokol HTTP yaitu GET, POST, PUT dan DELETE. Apabila method yang digunakan di luar dari yang sudah ditentukan maka server akan mengirimkan piggybacked response dengan kode 405 (Method not allowed).

Table 2.1 Method dalam protocol CoAP

Method	Keterangan
GET	Digunakan untuk mengambil informasi dari suatu resource berdasarkan URI yang diakses. Apabila dalam sebuah request terdapat opsi Accept maka server akan berusaha mengirim respon sesuai format yang diminta. Status respon dari method GET haruslah salah satu dari 2.xx (Success), 2.05 (Content) atau 2.03 (Valid).
POST	Digunakan untuk mengirimkan perintah pada server, tergantung pada URI dan resource yang hendak diakses. Biasanya method ini akan membuat atau mengubah nilai dari suatu resource. Apabila resource baru dibuat maka respon yang dikembalikan haruslah 2.01 (Created) beserta URI untuk mengakses resource tersebut. Apabila tidak membuat tetapi mengubah nilai sebuah resource maka kode respon haruslah 2.04 (Changed). Apabila sebuah resource dihapus, respon kodenya adalah 2.02 (Deleted).
PUT	Digunakan untuk mengubah nilai resource berdasarkan URI yang diakses. Apabila resource berhasil diubah maka kode respon yang

	diberikan harus 2.04 sedangkan apabila tidak berhasil maka server akan merespon dengan kode error yang sesuai.
DELETE	Digunakan untuk menghapus resource berdasarkan URI yang diakses. Method ini akan mengembalikan respon dengan kode 2.02.

2.2.5.5 CoAP URI

CoAP menggunakan coap dan coaps skema URI untuk mengidentifikasi resource yang hendak diakses. Skema lengkap CoAP URI adalah : "coap:" "//" host [":" port] path-abempty ["?" query] Apabila host adalah IP address maka CoAP server dapat diakses langsung menggunakan IP tersebut, namun apabila host yang digunakan adalah nama/domain, maka nama tersebut harus diubah menjadi alamat IP menggunakan DNS dan barulah server dapat diakses. Apabila bagian ini kosong, maka URI dianggap tidak valid. Bagian port mengindikasikan dimana port UDP server bekerja, apabila dikosongkan maka akan digunakan port default yaitu 5683.

Path menunjukkan resource yang hendak diakses. Path dapat terdiri dari beberapa segmen yang dipisahkan oleh karakter garis miring (U+002F SOLIDUS "/"). Query merupakan parameter tambahan untuk mengakses resource, dapat terdiri dari beberapa segmen yang dipisahkan dengan tanda (U+0026 AMPERSAND "&"). Query biasanya berbentuk pasangan “key=valule”.

2.2.6 Sistem Kualitas Air

Merupakan perancangan sebuah sistem yang telah banyak digunakan dalam penelitian-penelitian. Biasanya dibutuhkan sebagai studi kasus dalam penelitian yang bersifat implementasi atau pengembangan. Dalam penerapannya pun, sistem monitoring kualitas air banyak memiliki permasalahan di berbagai sector. Seperti pada ruang lingkup industri pertambakan. Industri pertambakan sangat memerlukan pemantauan kualitas air. Terlebih jika kualitas air tersebut sangat berpengaruh pada hasil budidaya tersebut. Meskipun demikian, tambak-tambak yang menggunakan sistem pemantauan otomatis hanya dalam lingkup tambak berskala besar. Dalam lingkup menengah masih sedikit yang melakukan penerapan tersebut.

Selain dalam bidang pertambakan, terdapat pula dalam bidang pertanian. Pertanian yang pada dasarnya juga sangat memerlukan sebuah pemantauan kualitas air sangat berpengaruh bagi kualitas hasil tani tersebut. Penelitian untuk bidang ini lebih cenderung dalam penerapan sebuah tindakan otomatis. Seperti sistem irigasi maupun pendeteksian kekeringan. Namun ada pula bidang pertanian yang membutuhkan pemantauan kualitas air. Seperti penerapan sebuah sistem pertaniam *aquaponic* atau kata lainnya hydroponic.

Kebutuhan suatu sistem pemantauan kualitas air dilihat dari penerapan penggunaan sensor-sensor yang dipilih. Sebagaimana untuk kebutuhan nilai

indikator kualitas air. Contohnya seperti kandungan kadar ph, apakah normal, asam atau basa. Kualitas suhu air maupun kandungan kadar garam. Bahkan ada yang menggunakan pendeteksian kadar oksigen dan tingkat kejernihan air. Berikut ini akan dijelaskan teori pendukung sebagai kebutuhan dalam implementasi yang melakukan studi kasus pada sistem pemantauan kualitas air.

2.2.6.1 Raspberry Pi

Raspberry Pi adalah mikrokomputer yang menggunakan mikroprosessor sebagai CPU utamanya (Horan, 2013). Raspberry Pi dilengkapi dengan prosesor ARM1176JZF-S 700 MHz, RAM sebesar 1 Gb dan sebuah GPU VideoCoreIV. Untuk penyimpanan data termasuk *Operating System*, Raspberry Pi tidak menggunakan Harddisk namun menggunakan SD Card. Raspberry Pi terdiri dari 2 versi yaitu Raspberry Pi model A dan Raspberry Pi model B. Untuk model A dan B memiliki spesifikasi yang sama dalam hal SoC, GPU, CPU SDRAM, *Output Video* dan *Audio*. Yang membedakan adalah pada model B memiliki 2 buah USB, sedangkan pada model A hanya memiliki 1 USB. Raspberry Pi juga dilengkapi dengan slot RJ45 (10/100 Ethernet) yang digunakan untuk jaringan internet. Dan untuk daya, kedua model ini memiliki daya yang berbeda, model A memiliki daya 300mA (1,5 W) dan model B 700mA (3,5 W).

Gambar 2.13 Raspberry Pi 2 Model B+
sumber: (www.raspberrypi.org)

Gambar 2.13 merupakan gambar dari Raspberry Pi 2 model B, mikrokomputer ini telah digunakan dalam penelitian pemrosesan data, pengiriman data dan jaringan. Karena fitur yang lengkap dan mempunyai RAM 1Gb namun ukurannya jauh lebih kecil, maka pada penelitian ini akan digunakan Raspberry Pi sebagai mikrokomputer yang berfungsi untuk memroses dan mengolah data pengolahan citra digital serta bertugas untuk mengirim data ke server. Pada Raspberry Pi 2 model B terdapat 4 port usb yang dapat digunakan untuk mengoperasikan webcam, serta daya yang dibutuhkan untuk operasi sangatlah kecil yaitu 5V dan arus minimal 1.2A.

Gambar 2.14 Raspberry Pi Zero
sumber: (www.raspberrypi.org)

2.2.6.2 Sensor PH

Ph merupakan suatu tingkat keasaman suatu benda dengan tingkat ke asaman skala ph antara o sampai 14, ph air yang bias di katakan normal bekisar antara 6,5 sampai dengan 8,5. Prinsip kerja pada sensor Ph air terletak pada sensor probe yang merupakan electrode kaca yang berfungsi mengukur jumlah ion H3O+ di dalam sebuah larutan, ujung pada alat ini merupakan sebuah kaca setebal 0,1mm yang berbentuk bulat (bulb). Bulb ini di pasangkan dengan sebuah silinder kaca nonkonduktor atau plastik memanjang yang selanjutnya di isi dengan larutan HCL.

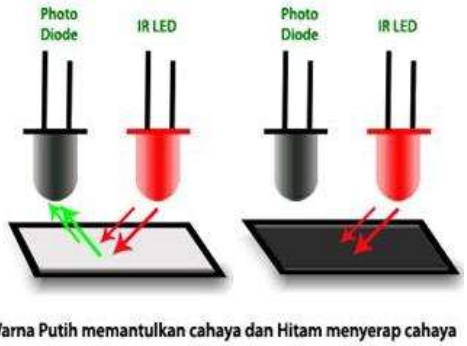
Gambar 2.15 Modul dan Probe Sensor PH
sumber: (probots.co.in)

Di dalam larutan HCL tersebut terendam sebuah kawat electrode panjang berbahan perak yang permukaanya terbentuk senyawa setimbang AgCl. Inti dari sensor ph tersebut terletak pada permukaan bulb kaca yang memiliki kemampuan untuk bertukar ion positif dengan larutan terukur, kaca tersusun atas molekul silicon dioksida dengan sejumlah ikatan logam alikali.

2.2.6.3 Sensor Photodioda

Sensor Photodioda merupakan suatu jenis dioda yang resistansinya berubah-ubah jika cahaya yang di terima oleh didao berubah intensitasnya, dalam kondisi gelap nilai tahanan sangat besar hinga membuat tak ada arus yang mengalir. Photodioda terbuat dari bahan semikonduktor berupa silicon atau gallium arsenide dan bahan lainnya, bahan-bahan ini menyerap cahay menyerap cahaya

melalui karakteristik jangkauan panjang gelombang, contoh :250nm ke 1100nm untuk silicon dan 800nm ke 2,0µm untuk GaAs. Photodioda di buat semikonduktor berbahan silicon untuk menyerap cahaya dengan panjang gelombang mencakup 2500Å – 11000 Å, ketika satuan energi dalam sumber cahaya diserap hal tersebut membangkitkan suatu electron dan menghasilkan sepasang pembawa muatan tunggal.

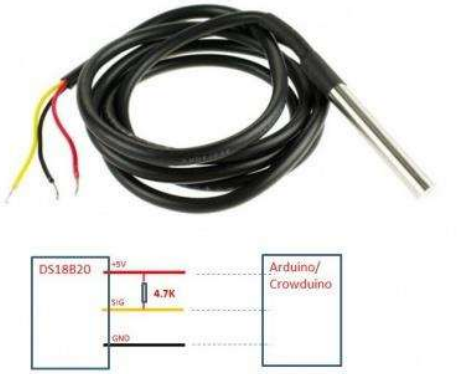


Gambar 2.16 Pembacaan LED dan Photodioda
sumber: (Wicaksono, 2011)

Saat photodiode terkena cahaya, maka photodiode akan bersifat sebgai sumber tegangan dan nilai resistansinya menjadi kecil, dan sebaliknya jika photodiode tidak terkena cahaya maka nilai resistansinya akan diasumsikan tak terhingga. Besar kecilnya tegangan atau arus listrik yang di hasilkan oleh photodiode tergantung besar kecilnya radiasi yang di pancarkan oleh infrared, jika photodiode bersambungan p-n maka akan bertegangan balik di sinari. Tanggapan frekuensi photodiode tidak terlalu luas, dari rentang gelombang yangdi hasilkan hanya sekitar 0,9µ.

2.2.6.4 Sensor Suhu (DS18B20)

Sensor adalah sesuatu yang digunakan untuk mendeteksi adanya perubahan lingkungan fisik atau kimia. Sensor suhu merupakan jenis sensor yang digunakan untuk mendeteksi setiap perubahan suhu.



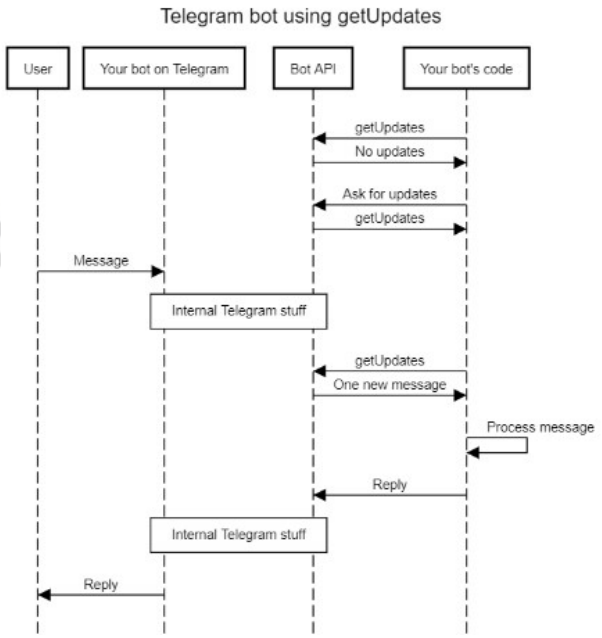
Gambar 2.17 Sensor Suhu DS18B20
sumber: (Makerlab Electronics)

Pada perancangan sistem ini saya menggunakan sensor suhu DS18B20 adalah digital thermometer, dimana keluaran sensor berupa data digital berupa sinyal pulsa yang mengindikasikan suhu tertentu. Sensor DS18B20 adalah jenis biasa dan waterproof. Range DS18B20 dari -55°C hingga 125°C. Sensor DS18B20 Pin-pinnya adalah pin 1 = output,pin 2 = 5V,pin 3 = GND. Sensor suhu DS18B20 dapat dilihat pada Gambar

2.2.7 Telegram API

Telegram adalah sebuah aplikasi pesan singkat seperti WhatsApp, Facebook dan WeChat. Mendapatkan popularitas dalam beberapa tahun terakhir karena berbagai alasan: *non-profit nature*, mendukung penggunaan *cross-platform*, fitur keamanan pesan yang terjamin, serta *open-source* API untuk pengembangan. Telegram API lebih dikenal sebagai Bot API. Berbasis HTTP, API untuk *developer* terhubung dengan *bot platform*. Bot API mengizinkan *developer* untuk mengatur Telegram bot. sebagai contoh ketika menerima pesan dan membalas pesan ke *user* yang berbeda.

Selain Bot API, terdapat juga Telegram API itu sendiri. API itu sendiri digunakan sebagai aplikasi Telegram untuk semua tindakan di dalam Telegram. Semisal, melihat chat, mengirim dan menerima pesan, mengubah *profil picture* atau membuat grup baru. Dengan menggunakan Telegram API, bisa melakukan pengembang apapun dengan aplikasi Telegram yang tetap berorientasi pada user *user friendly* (Yi, 2017).



Gambar 2.18 Method GET Telegram Bot
sumber: (towardsdatascience.com)

Metode bot API Telegram pada dasarnya adalah RPC (Remote Procedure Calls) melalui protokol HTTP. Fungsi ataupun library yang tersedia dapat ditemukan dalam dokumentasi Bot API. Setiap *methods* memiliki nama dan beberapa

parameter. Fungsi yang dipanggil oleh yang membuat permintaan HTTP yaitu parameter menuju ke API. Kemudian API *endpoint* menyesuaikan kepada bot dan fungsi yang dipanggil. Baik metode GET maupun POST, permintaan dapat diterima dengan baik. Karena parameter dapat dikirim sebagai JSON atau *payload* format-encode atau bahkan tipe data string itu sendiri.

2.3 Integration Testing

Integration testing dilakukan untuk menguji interaksi antara dua atau lebih komponen dalam software untuk menjalankan satu fungsi tertentu. Hasil dari penelitian ini ditujukan untuk mengetahui kesesuaian kebutuhan fungsional sistem, mengidentifikasi kesalahan pada komponen, serta dapat juga digunakan untuk pertimbangan perancangan dan kebutuhan non-fungsional. *Integration testing* dapat dibagi menjadi 4 model yakni Top-down, Bottom-up, Big-bang dan Sandwich.

Pada model Top-down pengujian dilakukan pada komponen pada tingkatan atas terlebih dahulu dilanjutkan oleh komponen pada tingkatan bawahnya. Apabila komponen pada tingkatan bawah dibutuhkan untuk pengujian komponen tingkat atas, komponen tersebut dapat diganti dengan *stubs*. Hal ini dapat mengurangi biaya pada saat pengujian. Model *Bottom-up* merupakan kebalikan dari *Top-down* dimana pengujian dimulai dengan komponen pada tingkatan paling bawah dilanjutkan dengan komponen di atasnya dan seterusnya hingga semua komponen teruji. Pada model *Big-bang*, setelah pengujian setiap komponen terkecil pada sistem selesai, selanjutnya semua komponen tersebut dijadikan satu kemudian diuji secara bersama-sama. Sedangkan model *sandwich* merupakan kombinasi antara *Top-bottom* dan juga Bottom up dalam *integration testing*.

2.3.1 Pengujian Sistem Kualitas Air

Air merupakan sumber daya alam yang berperan penting dalam kehidupan manusia. Salah satunya adalah untuk dikonsumsi. Air yang digunakan untuk konsumsi harus bersih, tidak berbau, berasa, berwarna dan sesuai standar yang telah ditetapkan oleh Kementrian Kesehatan. Alat ukur kualitas air ini menggunakan parameter suhu, kekeruhan, TDS, dan pH. Hasil akhir dari pembuatan alat ini masing-masing sensor pengukuran memiliki variasi eror. Yaitu sensor suhu dengan nilai eror maksimal 5,4% dengan standar deviasi rata-rata 1,145. PH dengan eror 0,848% dan standar deviasi rata-rata 0,01. TDS dengan eror 0,97% dan standar deviasi rata-rata 6,69 (Amani & Prawiroredjo, 2016).

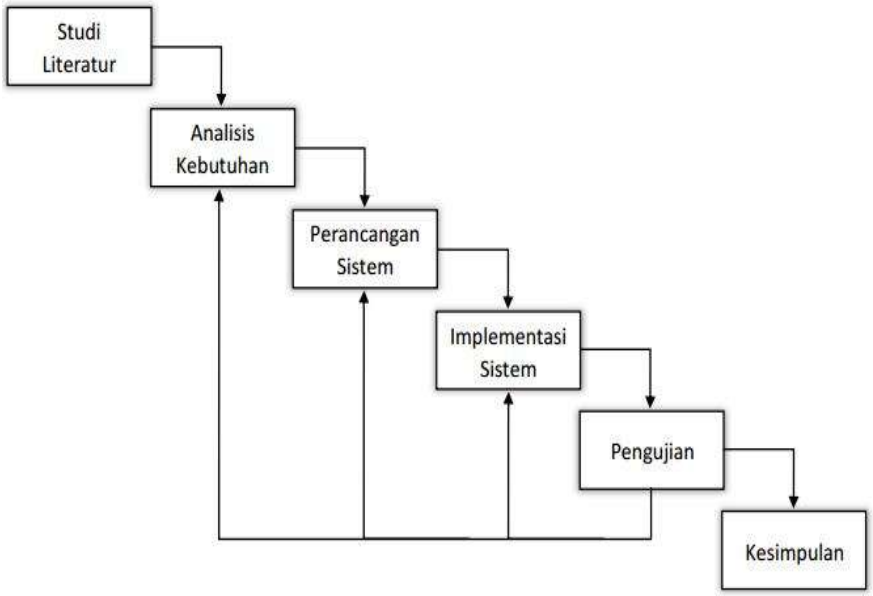
Untuk mendeteksi kualitas air minum, diperlukan standarisasi kualitas air dengan menggunakan parameter nilai suhu, kadar ph, dan tingkat kekeruhan menyesuaikan kebutuhan sistem yang diimplementasikan. Beberapa standarisasi kualitas air untuk keperluan konsumsi sebaiknya tidak asam atau basa, harus netral dalam parameter kadar pH. Nilai netral kadar pH adalah 7 sedangkan rentang nilai untuk air yang layak konsumsi adalah 6,5 s/d 8,5. Sedangkan untuk tingkat kekeruhan berada dalam rentang 5-20%. Dan untuk suhu, temperatur maksimum yang diperbolehkan adalah 3 derajat Celsius (Kesehatan, 2002).

BAB 3 METODOLOGI

Pada bab metodologi penelitian, dijelaskan langkah kerja dalam implementasi jaringan IPv6 dan protokol CoAP pada sistem *monitoring* kualitas air. Langkah-langkah yang dijelaskan disini meliputi studi literatur, analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian dan analisis serta pengambilan kesimpulan dan saran.

3.1 Metodologi Penelitian

Metodologi penelitian yang digunakan pada penelitian ini adalah metode penelitian *waterfall.* Metode tersebut merupakan metode pengembangan perangkat lunak terstruktur yang paling banyak digunakan secara luas, baik dalam lingkup akademisi maupun industri (Bintaro, 2015). Mengambil dasar dari metode penelitian *waterfall* menurut versi (Sommerville, 2011), penulis mengadaptasikannya pada metode penelitian tersebut dengan merubah dan menambahkan beberapa bagian sehingga dapat disesuaikan dengan alir penelitian penulis yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Metode Penelitian yang Digunakan

3.1.1 Studi Literatur

Studi literatur merupakan kegiatan yang dilakukan untuk mempelajari serta memahami penjelasan dasar teori yang digunakan untuk menunjang perancangan agar tidak mengalami kendala. Pada tahap studi literatur ini mempelajari teori-teori yang digunakan dalam pengerjaan skripsi. Teori-teori pendukung tersebut diperoleh dari buku, jurnal, e-book, dokumentasi, dan penelitian sebelumnya yang berkaitan dengan skripsi. Referensi utama yang diperoleh dalam penulisan ini adalah forum-forum terkait dengan komponen dan metode utama yang diterapkan dalam skripsi ini, seperti forum diskusi pengguna RaspberryPi,

implementasi CoAP, fragmentasi dan *header* pada jaringan IPv6*, wireless module* yang digunakan dan jurnal penelitian terkait.

3.1.2 Analisis Kebutuhan

Rekayasa kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan dalam merancang sistem yang akan dibuat dan diuji. Analisa kebutuhan dilakukan dengan cara mengidentifikasi kebutuhan dari sistem dan peralatan yang terlibat didalamnya. Dalam kebutuhan sistem akan terjadi proses mengidentifikasi beberapa p erangkat yang digunakan seperti perangkat keras dan perangkat lunak. Dengan adanya pengidentifikasi, maka dapat mempermudah dalam mendesain pembuatan sistem.

3.1.2.1 Kebutuhan Perangkat Keras

Perangkat keras yang dibutuhkan dalam implementasi sistem dalam penelitian ini yaitu Raspberry Pi sebagai media untuk penerapan 6lowpan, baik itu pada node-sensor maupun border router. Dipilihnya Raspberry Pi Zero sebagai node-sensor karena memiliki ukuran yang jauh lebih kecil layaknya mikrokontroler yang banyak digunakan. Walaupun memiliki ukuran yang kecil, namun raspberry pi layaknya mini pc yang memiliki kemampuan komputasi sehingga memudahkan dalam penataan letak bersama sensor-sensor yang digunakan. Raspberry Pi Type B+ yang digunakan sebagai border router juga memiliki kemampuan yang sama dengan raspberry pi zero, namun lebih unggul dalam hal kecepatan proses dengan kemampuan memori yang sedikit lebih besar.

Dipilih raspberry pi tersebut sebagai border router dikarenakan memiliki jumlah port yang mendukung sebagai access point maupun router gateway, sehingga implementasi yang dilakukan dapat mengoptimalkan fungsional sistem. Dibutuhkan juga microSD card 16GB untuk node-sensor dan 32GB untuk border router sebagai tempat sistem operasi raspbian jessie dan semua perangkat lunak yang dibutuhkan. Perangkat penunjang lain yang digunakan dalam penelitian ini adalah USB Adapter yang bertindak sebagai wireless adapter dan MRF24j40MA sebagai wireless adapter untuk komunikasi antar node.

Perangkat keras lain yang dibutuhkan yakni sensor yang digunakan dalam perancangan sistem *monitoring* kualitas air antara lain: sensor ph untuk mengindentifikasikan kadar asam/basa dalam air, sensor suhu (DS18B20) untuk data suhu dalam air, sensor kekeruhan (TDS) sebagai pendeteksi tingkat keruhnya air. Sensor-sensor tersebut nantinya akan mengirimkan data hasil ke border router untuk ditampung dan diolah untuk ditampilkan kepada user. *Sensor-node* tersebut menggunakan power adapter dengan output 12V/2.0A. Perangkat terakhir yakni pengguna menggunakan *smartphone* untuk mengakses API Telegram agar dapat menampilkan data dari border router yang menampilkan melalui BOT Telegram.

3.1.2.2 Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk mengimplementasikan 6lowpan disetiap node yakni sistem operasi raspbian jessie, custom kernel linux version

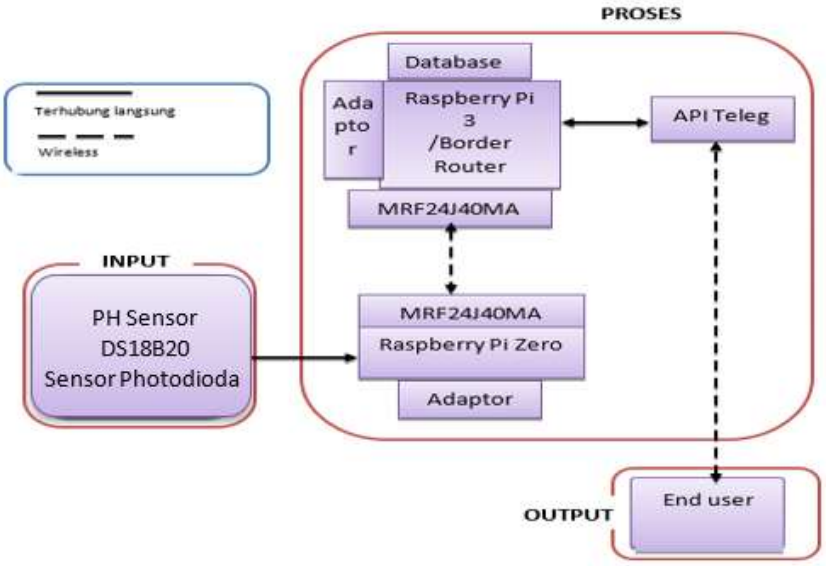
4.9.++, modul linux_wpan. Untuk implementasi prokol CoAP, perangkat lunak yang dibutuhkan yakni *aiocoap python* sebagai *library* CoAP python, asychronus python sebagai microservice yang terhubung dengan library CoAP. Sedangkan untuk penyimpanan data dari *sensor-node* menggunakan skema pengiriman langsung kepada pengguna menggunakan API Telegram dengan implementasi Chat BOT untuk kebutuhan data monitoring.

3.1.3 Perancangan Sistem

Perancangan sistem dalam penelitian ini terdiri dari perancangan alur sistem, perancangan *sensor-node*, perancangan 6LoWPAN *border router*, perancangan aplikasi, dan perancangan arsitektur jaringan untuk pengujian. Perancangan sistem dibuat dengan menggambarkan arsitektur sistem berdasarkan hasil analisis kebutuhan perangkat keras, kebutuhan perangkat lunak, kebutuhan fungsional dan kebutuhan non-fungsional yang telah ditentukan.Perancangan ini juga didasarkan pada teori yang diperoleh pada kajian pustaka berupa arsitektur middleware yang mendukung interoperabilitas. Perancangan sistem ini dibuat untuk mempermudah implementasi, pengujian adan analisis.

3.1.3.1 Perancangan Alur Sistem

Pada tahap perancangan, menjelaskan diagram alur kerja perancangan sistem yang akan diimplementasi pada monitoring kualitas air beserta perangkat yang dibutuhkan dengan diagram blok pada Gambar 3.3 yang dijelaskan terdapat tiga bagian perancangan sistem yaitu *input*, proses, dan *output.* Gambar 3.3 tentang blok diagram perancangan sistem yang terbagi menjadi yaitu *input*, proses, dan *output*.



Gambar 3.2 Diagram Blok Perancangan Sistem

Pada tahapan *input*, dilakukan pengambilan data oleh sensor yang telah ditentukan. Pada proses, terdapat 2 node, setiap node terdiri dari mikrokontroller atau mini komputer Raspberry Pi dan *wireless* modul MRF24J40MA. Node pertama berfungsi sebagai *sensor-node* yang menerima lalu meneruskan data

sensor menuju node lain yang difungsikan sebagai *border router* melalui hubungan nirkabel dengan mekanisme komunikasi menggunakan *protocol* CoAP. Pada tahapan *output* berupa data nilai kualitas air yang ditampilkan sesuai *parameter* masing-masing melalui aplikasi *chatbot*.

3.1.3.2 Perancangan Sensor-Node dan Node Border Router

Perancangan *sensor-node* yang dikembangkan pada penelitian ini menerapakan 6LoWPAN pada *sensor-node* sistem kualitas air yang terdiri dari sensor ph, sensor photodioda dan sensor suhu. Menggunakan arsitektur protokol pengiriman CoAP. Perancangan node border rrouter juga menerapkan arsitektuktur 6LoWPAN dan protokol CoAP sebagai media komunikasi pesan untuk menampung pesan terlebih dahulu dan nantinya akan diakses oleh user layaknya *middleware*. *Service unit* yang diterapkan disetiap *node* menggunakan MRF24J40MA sebagai *rf transmitter* dan *receiver*. *Application* gateway menyediakan interface bagi aplikasi untuk membaca data dari 6LoWPAN *border* router yang menyimpan data dari *sensor*-node melalui protokol CoAP. Perancangan ini akan dijelaskan lebih lanjut pada Bab 4.

3.1.3.3 Perancangan Aplikasi

Aplikasi dalam sistem digunakan sebagai *subscriber* yang akan membaca data dari border router dan juga untuk membuktikan bahwa data dari sensor dapat dikirimkan melalui *border router* yang dikembangkan. Aplikasi yang dibangun merupakan API Telegram berupa *chatbot* sederhana yang memiliki fungsi utama untuk menampilkan data dari *border router* dan merepresentasikannya dalam bentuk *graph*. Oleh karena itu aplikasi yang *chatbot* yang dikembangkan berbasis API Telegram, maka aplikasi ini dapat diakses dengan menambahkan akun *chatbot* ke dalam akun aplikasi Telegram yang terdapat di *smartphone*.

3.1.3.4 Perancangan Pengujian

Untuk mengetahui apakah sistem yang dibuat dapat bekerja sesuai dengan kebutuhan, maka dibutuhkan pengujian untuk membuktikannya. Dalam tahap ini terdapat dua jenis perancangan yakni perancangan topologi jaringan yang akan digunakan dan perancangan skenario pengujian. Pengujian dalam penelitian ini dibagi menjadi dua yakni *integration testing* dan pengujian sistem kualitas air. *Integration testing* digunakan untuk mengetahui apakah sensor-node dan border router yang dikembangkan menggunakan 6LoWPAN sudah sesuai dengan kebutuhan fungsional dari segi pengiriman data dan komunikasi antar *node*. Sedangkan pengujian sistem kualitas air dilakukan untuk menentukan apakah nilai ukur dari setiap parameter kualitas air telah memiliki nilai akurasi yang baik.

3.1.4 Implementasi

Implementasi sistem ini dilakukan berdasarkan dengan perancangan yang dibuat sebelumnya. Sistem ini akan mempunyai alamat *ipv6* yang dikonfigurasikan pada masing-masing nodenya didalam MRF24J40MA. Pada *sensor-node* difungsikan sebagai pengirim data sensor kepada *border router* untuk dilakukan

pengolahan data untuk dikirimkan kepada API Telegram yang menampung aplikasi *chatbot*. Dengan menggunakan MRF24J40MA, kedua node tersebut akan beroperasi pada *IPv6 over Low-power Wireless Personal Area Network* (6LoWPAN) dengan menggunakan *IEEE 802.15.4*.

Adapun *protocol stack* dari *6lowpan* yang diimplementasikan untuk membantu dalam fungsionalitas sistem agar dapat berjalan dengan sebagai mestinya, dijabarkan pada Tabel 3.1.

Tabel 3.1 6LoWPAN Protocol Stack

Layer	Implementasi
Application	CoAP (*Contrained Application Protocol*)
Transport	UDP (*User Datagram Protocol*)
Network	IPv6 (*Internet Protocol version 6*)
Adaptation	6LoWPAN (*Ipv6 over Low power Wireless Personal Area Network*)
Physical and Data Link	*IEEE 802.15.4*

Dari implementasi sistem ini, diharapkan mendapatkan hasil sebagai teknik untuk pengaturan jalur komunikasi antara *machine-to-machine* sehingga didapatkan data yang dibutuhkan dari sistem yang diterapkan.

3.1.5 Pengujian dan Analisis

Pengujian dari penelitian ini yaitu dengan memastikan data yang dikirim efisien secara *update* dalam skala waktu yang ditentukan untuk memperoleh hasil data yang diinginkan. Ketika data yang diterima pada *server* dam untuk selanjutnya diteruskan kepada pengguna ketika pengguna sedang membutuhkan data tersebut. Dengan menggunakan *parameter* pengujian yang sesuai pada perancangan sistem agar dapat berjalan baik sesuai yang diharapkan, pengujian sistem dilakukan meliputi :

1. Pengujian performa pengiriman data dan komunikasi antar *node.*
   Menerapkan pengujian dengan parameter delay terhadap jarak pengiriman.
2. Pengujian tingkat akurasi hasil ukur data sensor.

Melakukan perbandingan antara standarisasi kualitas air dengan hasil penerimaan data sensor yang diterapkan untuk mendapatkan beberapa nilai ukur kualitas air.

3.1.6 Kesimpulan

Pengambilan kesimpulan dilaksanakan dan didasarkan pada kesesuaian antara teori dengan penerapan. Setelah semua tahapan perancangan implementasi dan pengujian selesai, kesimpulan diuraikan untuk menjawab rumusan masalah yang didefinisikan pada penelitian. Tentang bagaimana mengimplementasikan jaringan IPv6 dan Constrained Application Protocol ke dalam sistem monitoring kualitas air berdasarkan kadar ph, suhu dan tingkat kekeruha. Tahapan terakhir dari penulis yang merupakan pemberian saran dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi selama proses penulisan dan implementasi jaringan 6lowpan. Serta memberikan pertimbangan dari rumusan topik-topik yang didapatkan atas hasil, untuk digunakan sebagai penelitian berikutnya.

BAB 4 REKAYASA KEBUTUHAN

Pada bab ini membahas gambaran umum sistem yang menjelaskan kebutuhan fungsional dan fitur yang ada pada penerapan 6LoWPAN dan protokol CoAP pada sistem *monitoring* kualitas air. Diuraiakan juga tentang rancangan arsitektur yang diterapkan agar perangkat sensor, *border router*, dan aplikasi saling berkomunikasi satu sama lain. Bab ini juga menjelaskan perancangan lingkungan penelitian yang berupa topologi jaringan serta metode dan skenario pengujian.

4.1 Deskripsi Umum Sistem

Penelitian ini merancang sebuah sistem sebagai lingkungan pengujian untuk implementasi IPv6 dan CoAP. Sistem ini terdiri dari dua *node* yakni 6LoWPAN *sensor-node* dan 6LoWPAN *border router*. *Sensor-node* akan mengirimkan data sensor berupa sensor kualitas air. Yang terdiri dari sensor suhu, kadar ph dan tingkat kekeruhan. Ketiga parameter kualitas air tersebut dikirim menuju *border router* dalam lingkup jaringan 6LoWPAN melalui protokol CoAP. *Border router* bertindak sebagai sensor *gateway* untuk *sensor-node* dengan mengirimkan *request message* untuk mendapatkan pesan data sensor dalam satuan waktu yang ditentukan. Kemudian *node border router* meneruskan pesan yang diterima menuju aplikasi *chatbot.* Yang difungsi untuk menampilkan data tersebut secara *real-time* ke dalam aplikasi pesan singkat Telegram.

4.1.1 Perspektif Sistem

Implementasi 6LoWPAN pada pengembangan *sensor-node* adalah menyediakan jaringan terbatas untuk kebutuhan komunikasi *machine-to-machine*. Dengan menyediakan *border router* berbasis 6LoWPAN bagi aplikasi untuk menyimpan hasil data yang dikirimkan *sensor-node* melalui protokol CoAP. Penerapan 6LoWPAN dan protokol CoAP ini berguna bagi pengembangan yang berfokus pada jaringan sensor nirkabel berkonsep perangkat IoT. Tanpa harus melihat protokol yang digunakan dan membebankan pada penggunaan atau alokasi *internet protocol v4* (IPv4).

Sistem ini berjalan dengan baik apabila sistem dapat melakukan pertukaran pesan menggunakan *interface* `lowpan0`. Yang mana merupakan kombinasi implementasi 6LoWPAN dan *module* MRF24J40MA disetiap *node* dengan protokol komunikasi yang diintegrasikan yaitu CoAP. Selain itu, *sensor-node* dapat membaca setiap nilai sensor untuk dikirimkan kepada *node border-router* yang setelah itu hasil pembacaan nilai sensor dilakukan analisis. Untuk ditampilkan kepada user sebagai kebutuhan hasil monitoring kualitas air.

4.1.2 Karakteristik Pengguna

Karakteristik pengguna pada sistem ini hanya bersifat pasif, yaitu pengguna hanya bisa melihat data nilai sensor untuk kebutuhan monitoring. Data yang dihasilkan merupakan analisis dari hasil perbandingan standarisasi kualitas air dengan nilai pembacaan sensor yang dikirimkan oleh *sensor-node*. Untuk

pengaturan waktu dalam memilih hasil data dapat dilakukan oleh pengguna. Dengan menerapkan penggunaan API disalah satu aplikasi *mobile* yang mendukung pengembangan *chatbot* sesuai kebutuhan, diharapkan hasil data monitoring dapat efektif diterima oleh pengguna

4.1.3 Batasan Sistem

Beberapa batasan yang diterapkan ada sistem ini antara lain:

1. Sensor yang digunakan adalah sensor suhu (*waterproof*), sensor ph, dan sensor kekeruhan air.
2. Setiap sensor akan melakukan pembacaan nilai setiap 20 menit secara bersamaan.
3. Jarak yang diterapkan antar *node* maksimal 50 meter (setelah dilakukan pengujian jarak pengiriman pesan terhadap node).
4. Sumber tegangan untuk setiap *node* merupakan sumber tegangan langsung melalui adaptor dengan tegangan minimal 5v, belum diterapkan sumber tegangan yang langsung dibutuhkan oleh *node* agar bisa dilakukan pengujian dengan parameter konsumsi daya (*green smart energy*).

4.1.4 Asumsi dan Ketergantungan

Beberapa asumsi dan ketergantungan yang ada pada sistem ini antara lain:

1. Setiap sensor akan diterapkan untuk melakukan pembacaan nilai sensor setiap 20 menit sekali secara bersamaan.
2. Penerapan pengiriman pesan dengan *messages model request and response* yang membuat *sensor-node* harus tetap menerima *request* dari *node-border router* ketika terjadi gangguan atau *non-confirmable* status dari *sensor-node*.
3. Tegangan 5V merupakan yang paling aman untuk setiap sensor dan *board* raspberry pi yang digunakan.
4. Penggunaan ULA (*Unique Local* Address) *ipv6* pada setiap *node* agar dapat dilakukan *global routing* sebagai pengalamatan *interface lowpan* yang digunakan.

4.2 Kebutuhan Fungsional

Kebutuhan fungsional adalah kebutuhan-kebutuhan serta proses yang harus dipenuhi dan dapat dilakukan oleh sistem. Kebutuhan sistem dalam penelitan ini dijelaskan pada Tabel 4.1 berikut ini:

Tabel 4.1 Kebutuhan Fungsional Sistem

No	Kebutuhan Fungsional
1	RaspberryPi dapat menyediakan layanan *6lowpan sensor-node* dan *6lowpan border router*.
2	MRF24J40MA disetiap *node* dapat mengaktifkan protokol jaringan *IEEE 802.15.4* (`lowpan` dan `wpan`).
3	*6lbr* dapat menyediakan layanan gateway bagi *sensor-node*.

4	*6lowpan sensor-node* dapat terhubung dengan *6lowpan border router* melalui protokol CoAP.
5	*Sensor-node* dapat menyediakan *resource* protokol CoAP untuk *message* model confirmable.
6	*Sensor-node* dapat mengirimkan data kualitas air ke *6lowpan border routerr* melalui protokol CoAP.
7	*Border router* dapat menerima data kualitas air sebagai payload dari *sensor-node* melalui protokol CoAP.
8	*Border router* dapat mengirimkan data kualitas air menuju API Telegram untuk ditampilkan pada akun *chatbot* yang didalam aplikasi Telegram.
9	Chatbot dapat menampilkan data kualitas air yang dikirimkan *sensor-node* berbasis jaringan internet.

Kebutuhan fungsional merupakan penunjang kinerja sistem dan sangat diperlukan dalam sebuah perancangan dan implementasi. Agar dapat bekerja dengan baik seperti yang diharapkan.

1. Sistem dapat menampilkan hasil baca sensor pada *web-app* yang telah dirancang, dimana data tersebut telah melalui serangkaian skema pengiriman data dari *sensor-node* menggunakan protocol CoAP dengan baik.
    a. Penjelasan
       Fitur ini berguna untuk menjelaskan bagaimana pengguna dapat mengakses hasil dari penerimaan data sensor untuk dapat diketahui status kualitas air. Proses tersebut adalah pengguna hanya cukup menambahkan akut *chatbot* yang telah disediakan melalui jaringan local yang berada dalam satu lingkup jaringan dengan *node border router*. Kemudian akan langsung menampilkan tabel hasil monitoring dengan parameter penilaian yang digunakan.
    b. Stimulus atau respon sistem
       Sistem akan berjalan ketika *sensor-node* mulai diaktifkan dengan status menunggu *request message* dari *node border router*. Ketika *node border router* diaktifkan dan melakukan *requestGET*, maka akan mendapatkan *response message* berupa nilai sensor yang kemudian ditampilkan pada halaman akun *chatbot* Telegram.
    c. Kebutuhan fungsional
       Kebutuhan fungsional ini digunakan sebagai cara agar pengguna dapat melihat hasil monitoring yang dilakukan.
2. Hasil data yang diterima merupakan hasil olah untuk menunjukkan bagaimana keadaan kualitas air sesuai dengan parameter penilaian yang digunakan serta dapat memastikan waktu dan tanggal dari data sensor yang diperoleh.
    a. Penjelasan

Fitur ini berguna untuk menjelaskan hasil penerimaan data sensor yang awalnya hanya berupa nilai-nilai. Hasil tersebut kemudian dikonversikan kedalam status keaadan dengan keakurasian hasil uji sensor-sensor tersebut berikut dengan waktu dan tanggal yang diperoleh akan ditampilkan pada aplikasi *chatbot* sebagai indikasi waktu nyata.

b. Stimulus atau respon sistem
Node border router merupakan yang bertugas dalam hal ini. Dilakukan proses untuk mengkonversikan kedalam status dan kategori agar selanjutnya ketika diakses pada API tujuan sudah menjadi data yang bisa dipergunakan oleh pengguna bukan berupa data metah. Terdapat pengaturan delay pada pengiriman data sensor yang pertama menuju data sensor selanjutnya. Hal ini diterapkan bertujuan agar tidak terjadinya penumpukkan data yang tidak perlu karena dalam pemantuan kualitas air tidak terlalu diperlukan penerapan *real-time*.

c. Kebutuhan fungsional
Kebutuhan fungsional ini merupakan cara agar pengguna dapat membaca hasil data monitoring sebagai data teknis.

4.3 Kebutuhan Antarmuka

Kebutuhan antarmuka merupakan kebutuhan pengguna sistem untuk terhubung dengan sistem itu sendiri. Kebutuhan antarmuka pada bab ini dikategorikan menjadi tiga bagian, yakni kebutuhan antarmuka pengguna, kebutuhan perangkat keras serta kebutuhan perangkat lunak.

4.3.1 Kebutuhan Antarmuka Pengguna

Dalam sistem ini, kebutuhan yang diperlukan untuk antarmuka pengguna adalah sebagai berikut:

1. Sistem dapat menampilkan hasil baca sensor pada salah satu aplikasi *chatbot* yang telah dirancang, dimana data tersebut telah melalui serangkaian skema pengiriman data dari *sensor-node* menggunakan protocol CoAP dengan baik.
2. Hasil data yang diterima merupakan hasil olah untuk menunjukkan bagaimana keadaan kualitas air sesuai dengan parameter penilaian yang digunakan.
3. Pengguna juga dapat memastikan waktu dan tanggal dari data sensor yang diperoleh.

4.3.2 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras untuk implementasi 6LoWPAN sistem kualitas air dalam penelitian ini yaitu:

Tabel 4.2 Kebutuhan Perangkat Keras

Perangkat	Keterangan
Raspberrypi Zero	Digunakan sebagai perangkat untuk implementasi sistem pemantauan kualitas air berbasis *6lowpan*.

Raspberrypi B	Digunakan sebagai perangkat *border router* berbasis *6lowpan*.
MRF24J40MA	Digunakan sebagai *wireless module* untuk mengaktifkan *wpan* (*IEEE 802.15.4*)
USB Adapter TP-LINK	Perangkat ini dibutuhkan sebagai pendukung *6lbr* raspberry pi agar dapat menyediakan *access point*.
Sensor PH	Modul sensor untuk membaca kadar asam/basa pada air.
Sensor DS18B20	Modul sensor *waterproof* untuk membaca nilai suhu air.
Sensor Turbidity	Modul sensor untuk membaca tingkat kekeruhan air.
Laptop	Perangkat ini digunakan untuk mengakses *web-app* dalam hal menampilkan data dari *6lbr* dan digunakan untuk melalukan konfigurasi raspberry pi melalui *remote-access*.

4.3.3 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang dibutuhkan untuk mengimplementasikan *6lowpan* pada *node* dan sistem kualitas air serta protocol komunikasi CoAP dalam penelitian ini yaitu:

Tabel 4.3 Kebutuhan Perangkat Lunak

Perangkat	Keterangan
Raspbian Jessie	Sistem operasi untuk raspberry pi.
Linux kernel 4.7++	Kernel linux untuk implementasi *6lowpan.*
raspi_wpan (github)	Modul *6lowpan* yang diperoleh dari *github*.
Python version 3	Pemrograman untuk protokol komunikasi yang digunakan.
Asyncio python	Framework python untuk *asyncronus*.
aiocoap	Library CoAP berbasis asyncio python.
SSH	Remote access yang dilakukan terhadap raspberry pi untuk pengkonfigurasian.
Telegram Chatbot	Menampilkan data sensor yang telah diperoleh dari *sensor-node*.

4.4 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional dalam penelitian ini menggunakan *access point* atau *router* yang berada pada lingkup jaringan local. Dengan menghubungkan ke

port LAN yang tersedia dan pengguna terkoneksi kedalam satu lingkup jaringan tersebut, pengguna akan bisa mengakses aplikasi *chatbot* didalam Telegram untuk mendapatkan hasil monitoring yang dilakukan sistem.

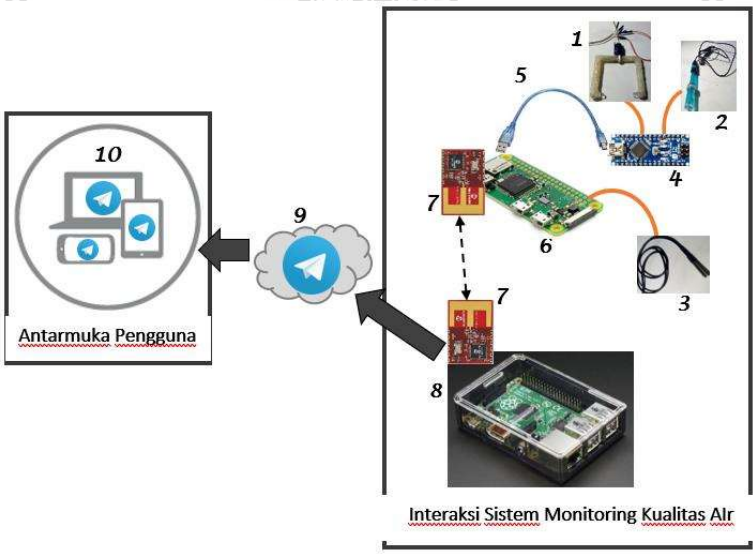
Selain itu, untuk masing-masing *node* bisa menggunakan sumber tegangan langsung maupun menggunakan *powerbank*, karena penerapan dari sistem ini memiliki tujuan sebagai *low power wireless* sehingga tegangan yang diperlukan tidak terlalu besar agar sistem dapat berjalan dengan baik. Ketika terdapat kesalahan dalam jalannya sistem, pengguna yang sebagai administrator dapat menjalankan sistem secara manual dengan melakukan *remote access* terhadap masing-masing *node*.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Bab ini menjelaskan tentang rancangan arsitektur yang diterapkan agar perangkat sensor, *border router*, dan aplikasi saling berkomunikasi satu sama lain dan perancangan lingkungan penelitian yang berupa topologi jaringan. Sedangkan proses implementasi rancangan arsitektur yang sudah dibuat menjadi *sensor-node* dan *node border router* pada mini komputer RaspberryPi dengan menampilkan *pseudocode* tentang proses-proses utama yang dibutuhkan oleh jaringan 6LoWPAN.

5.1 Gambaran Umum Sistem

Secara keselurahan sistem terdapat tiga komponen yang saling berinteraksi. Komponen pertama merupakan interaksi antar jaringan *node* 6LoWPAN. Interaksi tersebut merupakan proses pertukaran pesan antara *Sensor-Node* yang melakukan pembacaan nilai kualitas air dengan *Node Border-Router* sebagai penerima hasil kualitas air yang berupa data mentah. Jenis sensor kualitas air yang digunakan adalah sensor pendeteksi kadar pH, sensor pendeteksi tingkat kekeruhan dan sensor pendeteksi suhu pada air. Komponen kedua adalah interaksi antar jaringan. Jaringan yang dimaksud yakni jaringan sensor dan jaringan internet. Dalam interaksi ini dilakukan proses pengenalan terhadap alamat akun *chatbot* yang dikembangkan menggunakan API Telegram. Mulai dilakukan pengkonversian hasil sensor kualitas air menjadi status kondisi yang sesuai dengan standarisasi kualitas air. Komponen ketiga yakni interaksi pengguna sebagai *end-user* menggunakan aplikasi Telegram.



Gambar 5.1 Diagram Blok Interaksi Sistem

Keterangan pada Gambar 5.1 diatas adalah:

1. Sensor analog berupa *Photodioda* dan LED sebagai sensor kekeruhan.
2. Modul analog sensor pH sebagai sensor kadar pH air.

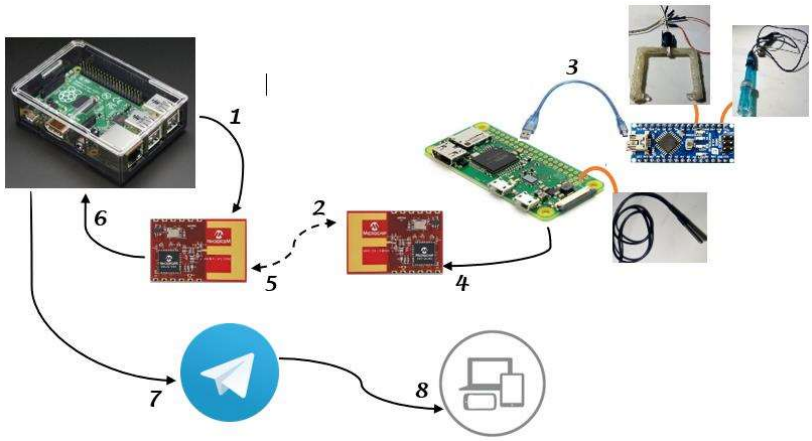
3. Sensor digital DS18B20 sebagai sensor suhu air.
4. Arduino Nano sebagai pemrosesan sensor analog.
5. USB Serial sebagai media pengiriman hasil baca sensor analog.
6. RaspberryPi Zero sebagai *Sensor-Node* pemroses semua hasil baca sensor.
7. MRF24J40MA sebagai modul *wireless* dengan lingkup jaringan 6LoWPAN.
8. RaspberryPi B+ sebagai *Node Border Router* menerima hasil baca sensor yang dikirimkan oleh *Sensor-Node*.
9. API Telegram sebagai penyedia layanan *chatbot*.
10. *Telegram User Chatbot* sebagai penerima hasil akhir kualitas air beserta statusnya.

Implementasi yang dilakukan untuk kebutuhan *Sensor-Node* terdiri dari sensor suhu, sensor ph, dan sensor tingkat kekeruhan berbasis 6LoWPAN dan CoAP. *Sensor-Node* mengirimkan nilai sensor sebagai parameter kualitas air ke *Node Border-Router*. Menggunakan skema pengiriman *response message* sebagai balasan dari *request message Node Border-Router*. Sebelum *Sensor-Node* dapat mengirimkan data, masing-masing *node* harus terhubung melalui *interface* `lowpan0` yang berada dalam lingkup jaringan 6LoWPAN menggunakan ULA IPv6 yang telah dilakukan untuk *global routing*.

5.2 Perancangan Sistem

Tahap perancangan dalam penelitian ini akan menjelaskan tahapan perancangan 6LoWPAN yang terdiri dari perancangan *sensor-node* dan perancangan *node border router*, perancangan *application gateway* serta perancangan topologi jaringan. Hal tersebut bertujuan agar sistem bekerja sesuai dengan perancangan dan implementasi yang dilakukan. Interaksi antar komponen dalam sistem dibagi menjadi dua yakni pengiriman data dari *sensor-node* ke *node border router* melalui protokol CoAP didalam lingkup jaringan 6LoWPAN dan pengolahan data yang diterima oleh *node border* router akan ditampilkan pada halaman akun *chatbot* untuk diakses pengguna.

Model dan ukuran data yang digunakan saat pengiriman ditentukan oleh pembuat program. Interaksi kedua yakni pengiriman data dari *node border router* ke Telegram *API*. Setiap kali sensor mengirimkan data, maka *chatbot* akan menerima data tersebut dengan pengaturan waktu pengiriman yang ditentukan. Perancangan alur komunikasi sistem 6LoWPAN dengan CoAP yang dikembangkan dapat dilihat pada Gambar 5.2.



Gambar 5.2 Alur Komunikasi Sistem

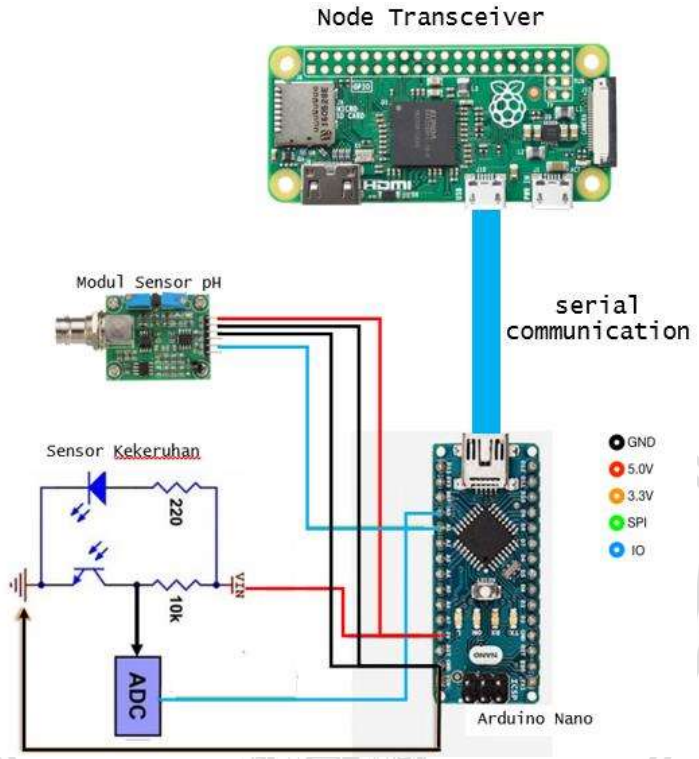
Keterangan Gambar 5.2.

1. Node *border router* mengirimkan *request message* kepada *sensor node* untuk mendapatkan nilai sensor kualitas air.
2. Request *message* yang diinisiasikan sebagai topik `CONTENT` dikirim menuju *interface* `lowpan0` milik *sensor-node* menggunakan ULA *global route* IPv6.
3. *Sensor-node* berstatus menunggu *request message* melakukan pembacaan sensor analog menggunakan *serial communication* terhadapan Arduino Nano.
4. Sensor node mengembalikan topik `CONTENT` berupa *response message* dikirimkan kembali melalui *interface* `lowpan0` menggunakan ULA *global route* IPv6 menuju interface `lowpan0` *gateway*.
5. Interface `lowpan0` *node border router* menerima aktifitas komunikasi melalui protokol CoAP dari *interface* `lowpan0` yang menuju titik *gateway*.
6. Node *border router* menerima *response message* dari *sensor-node* berupa `CONTENT payload`. *Response message* tidak disimpan ataupun ditampilkan, melainkan langsung diteruskan menuju API.
7. Node *border router* yang berperan sebagai *application gateway* mempunyai perintah `sending messages` kepada API Telegram yang dirancang untuk menampilkan hasil data sensor.
8. API Telegram yang telah terhubung dengan *node border router* menerima data sensor yang kemudian ditampilkan pada aplikasi Telegram dengan menambahkan user *chatbot* sebagai *broker*.

5.2.2 Perancangan Sensor Node

Perancangan perangkat keras pada *sensor-node* sebagai *transceiver* terdiri dari sensor analog dan sensor digital yang dirangkai menjadi satu. Sensor analog menggunakan Arduino Nano sebagai *converter* input analog menjadi digital.

Kemudian hasil konversi tersebut dikirimkan melalu *serial communication* menuju *node transceiver* agar dapat dilakukan pemrosesan Bersama hasil sensor digital. Terdapat beberapa komponen diantaranya Arduino Nano sebagai mikrokontroler untuk kebutuhan *Analog Digital Converter* (ADC), penggunaan pin sensor kekeruhan dengan *Photodioda* dan *LED*, port modul sensor *pH*, dan RaspberryPi Zero yang dihubungkan secara *serial*.



Gambar 5.3 Skematik Sensor Analog

Penjelasan dari rancangan penggunaan pin sensor kekeruhan terdapat pada Tabel 5.1 dibawah ini. Dengan membaca nilai tegangan dari penggunaan LED dan Photodioda dihubungkan dengan pin analog *input* Arduino Nano. Menerima nilai keluaran dari sensor agar dapat memproses nilai dan diolah sebagai nilai kekeruhan. Penggunaan pin GND dan Vin sensor dihubungkan dengan pin GND dan pin 5V Arduino Nano. Penerapannya sama dengan rangkaian sensor suhu hanya saja terdapat perbedaan pada pin I/O Arduino Nano terdapat pada pin A1 sebagai input analog. Dan menambahkan resistor untuk hubungan antara *Photodioda* dan *LED*.

Tabel 5.1 Keterangan Pin Sensor Kekeruhan

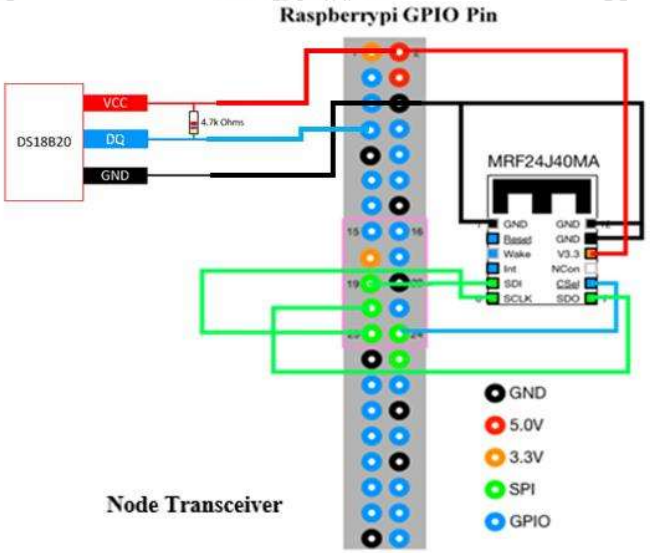
Pin Sensor *Photodioda*	Pin Arduino Nano
Vin	5V
I/O	A0
GND	GND

Tabel 5.2 Keterangan Pin Sensor *pH*

Pin Modul Sensor pH	Pin Arduino Nano
V++	5V
G	GND
G	GND
D0	A1

Tabel 5.2 merupakan keterangan pin modul sensor pH dengan Arduino Nano. Sensor PH memiliki satu pin I/O sebagai pin D0. Digunakan untuk terhubung dengan Arduino Nano pada pin A1 sebagai pembacaan nilai analog. GND dan VCC untuk modul sensor PH dan Arduino Nano terhubung dengan mekanisme yang sama dengan modul dan sensor kekeruhan. Namun didalam modul sensor *pH* terdapat 2 pin untuk *ground* yang disimbolkan sebagai G. Kedua pin G tersebut sama-sama terhubung dengan pin GND Arduino Nano. Inputan nilai yang diterima merupakan hasil kalibrasi antara *probe sensor* dengan PH-Kit. Kalibrasi tersebut dilakukan untuk mendapatkan *measurement value* kadar asam didalam air. Dilakukan perbandingan dengan nilai hasil inputan *probe* sensor dengan melakukan pengujian langsung pada air yang dipilih sebagai sampel.

Perancangan yang dilakukan untuk *node transceiver* terdiri atas sensor DS18B20 sebagai inputan nilai sensor digital untuk indikator suhu. Selain itu, terdapat pula RaspberryPi Zero yang digunakan sebagai mikroprocesor melakukan pemrosesan hasil baca sensor. Baik itu dari input sensor analog maupun sensor digital. Didalam RaspberryPi Zero terdapat perangkat keras modul *wireless* MRF24J40MA sebagai media komunikasi jaringan antar *node*. Penjelasan dari perancangan untuk *node transceiver* dapat dilihat pada gambar skematik dibawah ini.



Gambar 5.4 Skematik Sensor-Node Sebagai Transceiver

MRF24J40MA yang digunakan dalam penelitian ini merupakan RF *module* untuk komunikasi *wireless* dengan *mini pc* RaspberryPi Zero sebagai mikrokontroller. Modul MRF24J40MA memiliki total 12 pin yang terdiri dari 3,3 V pin VCC, RESET, WAKE, INT, SDI, SCK, NC, CS, SDO dan tiga pin GND. Pada Tabel 5.3 dibawah ini, merupakan penjelasan dari penggunaan pin pada modul MRF24J40MA. Agar dapat terhubung dengan Raspberry Pi. Pin GND modul MRF24J40MA yang terdapat pada pin 12, 11, dan 1. Ketiganya dihubungkan menjadi satu lalu dihubungkan dengan pin GND pada RaspberryPi.

Tabel 5.3 Keterangan Pin MRF24J40MA pada Sensor-Node

Pin MRF24J40MA	Pin Raspberry Pi Zero
5 (SDI)	19 (MOSI)
6 (SCK)	23 (CLK)
7 (SDO)	21 (MISO)
8 (CS)	24 (CEO)
10 (Vin)	2 dan 4 (Vin)
12, 11, dan 1 (GND)	6 (GND)

Sedangkan pada penelitian ini, RaspberryPi menggunakan pin 6. Untuk tegangan yang diberikan, melalui RaspberryPi dihubungkan pada pin 2 dan 4. Vin dengan menggunakan nilai tegangan maksimal 5V dan dihubungkan ke modul MRF24J40MA pada pin 10 Vin sebesar 3,3V. Hal tersebut dilakukan karena minimal tegangan yang bisa diberikan adalah 3,3V. Dan maksimal 5V. Sedangkan untuk pin SPI pada RaspberryPi yang terdiri dari MOSI, MISO, CLK dan CEO, dihubungkan pada pin SDI, SCK, SDO dan CS pada modul MRF24J40MA. SDI dan SDO modul MRF24J40MA terdapat pada pin 5 dan pin 7. Terhubung ke pin 19 dan 21 pada RaspberryPi sebagai MOSI dan MISO. Sedangkan SCK dan CS modul MRF24J40MA terdapat pada pin 6 dan 8. Terhubung ke pin 23 dan 24 pada RaspberryPi sebagai CLK dan CEO.

Sensor suhu DS18B20 memiliki pin yang terdiri dari pin GND, VCC dan I/O (input/output). Penerapan yang dilakukan pada penelitian ini membuat skematik rangkaian yang saling menghubungkan antara pin GND dan VCC. Setiap modul dan sensor yang digunakan dengan memberikan pilihan nilai tegangan mana yang akan digunakan. Antara tegangan 3,3V maupun 5V dapat dipilih dan disesuaikan. Untuk penggunaan pin I/O pada sensor suhu ini dihubungkan pada pin 7 GPIO RaspberryPi. Dengan kondisi penambahan resistor 10k antara VCC dan I/O pada pin sensor dilihat dari datasheet.

Tabel 5.4 Keterangan Pin Sensor Suhu DS18B20

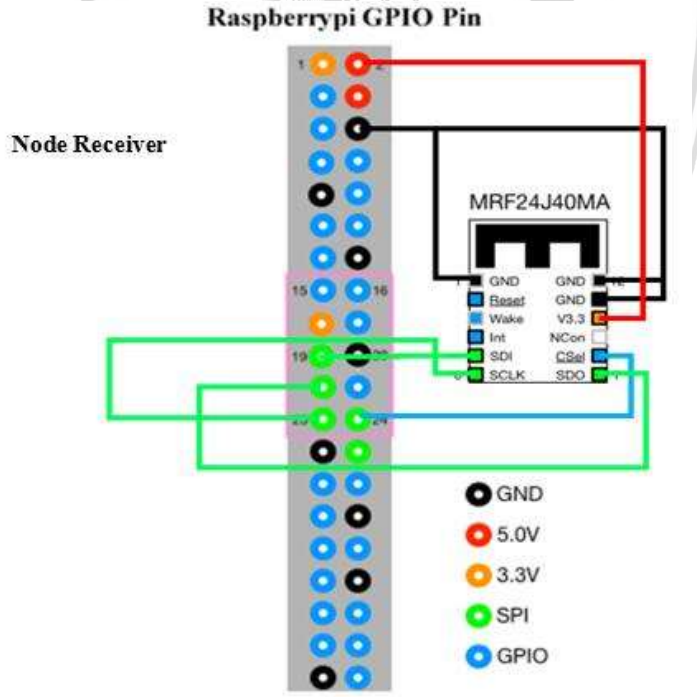
Pin Sensor DS18B20	Pin Raspberry Pi Zero
Vin	2 dan 4 (Vin)
I/O	7 (GPIO)

GND	6 (GND)

Untuk perancangan perangkat lunak *sensor-node* berupa pengkondisian sistem melalui pemrograman shell. Melakukan konfigurasi untuk mengaktifkan *interface* `wpan` dan `lowpan0` agar dapat menjalankan komunikasi *wireless.* Namun, sebelumnya harus dilakukan pemanggilan pada *library* MRF24J40MA yang terdapat didalam file `dtoverlay/*` RaspberryPi. Pengalamatan yang didapatkan pada *sensor-node* adalah mengakses *gateway address node border router*. Bertujuan agar dapat melakukan pengiriman pesan menggunakan *interface* `lowpan0` melalui jaringan *6lowpan* protokol *IEEE 802.15.4* melalui ULA *global route*.

5.2.3 Perancangan Node Border-Router

Node border router berbasis 6LoWPAN dalam penelitian ini dirancang agar komunikasi antar *node* dalam lingkup jaringan 6LoWPAN dapat terpenuhi. Perancangan 6LoWPAN *node border router* digunakan sebagai *receiver*. Dimana dilakukan konfigurasi *border gateway* sebagai pengalamatan. Hal tersebut bertujuan agar *sensor-node* dapat terhubung didalam lingkup jaringan yang sama dan dapat melakukan komunikasi untuk pertukaran pesan. Gambar skematik rangkaian ini dapat dijelaskan pada gambar dibawah.



Gambar 5.5 Skematik Node Border Router Sebagai Receiver

Pada Gambar 5.5 skematik perancangan *node border router* sebagai *receiver*. Kompoenen perangkat keras yang terdapat pada *node* tersebut yaitu Raspberry Pi Type B+ dan MRF24J40MA. Karena digunakan hanya untuk mengolah dan menyimpan sementara pesan. Pesan berupa data sensor yang diterima dari *sensor-node* kemudian diteruskan ke API Telegram untuk selanjutnya ditampilkan.

Penggunaan API Telegram bertujuan untuk menampilkan hasil data monitoring melalui akun *chatbot* yang bersifat *open-source*.

Tabel 5.5 Keterangan Pin MRF24J40MA pada Node Border Router

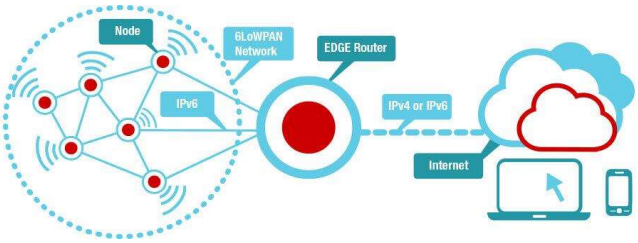
Pin MRF24J40MA	Pin Raspberry B+
5 (SDI)	19 (MOSI)
6 (SCK)	23 (CLK)
7 (SDO)	21 (MISO)
8 (CS)	24 (CEO)
10 (Vin)	1 (Vin)
12, 11, dan 1 (GND)	6 (GND)

Modul RF yang diimplementasikan didalam perancangan *node* ini sama dengan yang ada pada *sensor-node* yaitu MRF24J40MA. Penggunaan RaspberryPi Type B+ dipilih karena memiliki pin GPIO yang sama dengan RaspberryPi Zero. Selain itu terdapat *port ethernet* yang dapat digunakan untuk memberi akses kepada pengguna melakukan pemantauan hasil data *monitoring*. Tabel 5.5 diatas merupakan keterangan dari penggunaan pin untuk *node border router* antara MRF24J40MA dengan RaspberryPi. Penggunaan pin tersebut agar saling tersebut masih sama dengan apa yang diterapkan pada *sensor-node*. Hanya saja tegangan yang digunakan dalam perancangan ini menggunakan tegangan minimal yaitu 3,3V. Sedangkan port MOSI dan MISO pada pin SPI RaspberryPi tetap terhubung dengan port SDI dan SDO pada pin MRF24J40MA.

Perancangan perangkat lunak pada *node receiver* ini juga sama dengan *node-sensor*. Yakni, dalam hal *setup* konfigurasi untuk *interface* jaringan. Namun terdapat penambahan perintah fungsi untuk melakukan hubungan terhadap API Telegram. Di dalam *node* receiver, tidak hanya terdapat fungsi tambahan tersebut. Melainkan ada perintah fungsi lain untuk meneruskan pesan yang diterima. Perintah tersebut akan berperan ketika terhubung dengan API Telegram lalu menampilkan hasil ke dalam akun *chatbot*.

5.2.4 Perancangan 6LoWPAN dan CoAP

Dalam penelitian ini, penulis menggunakan metode 6LoWPAN sebagai jaringan untuk *sensor-node* dan *node border router*. Penerapan metode ini dilakukan untuk pembatasan lingkup komunikasi *machine-to-machine*. Yang bertujuan untuk dapat menekan penggunaan daya namun memiliki jangkauan yang terbatas. 6LoWPAN *network* bergerak dalam bagian IEEE 802.15.4. Dengan penggunaan *RF module* MRF24J40MA yang *compatible* dengan segala jenis mikrokontroler. Untuk lebih jelasnya mengenai lingkup jaringan ini, dapat dilihat pada gambar arsitektur dibawah ini.



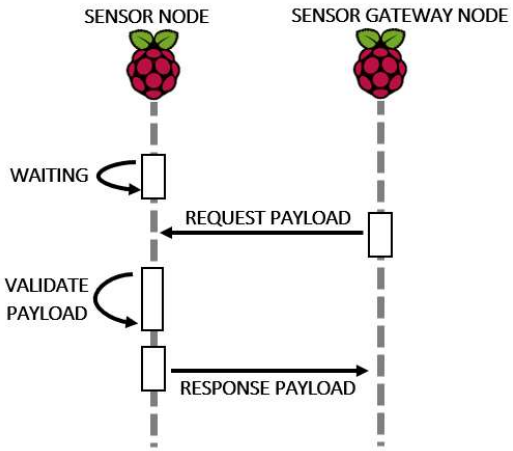
Gambar 5.6 Ruang Lingkup Jaringan 6LoWPAN
sumber: (Schmidt, 2016)

Arsitektur tersebut merupakan jaringan tersendiri yang dirancang. Agar dapat menunjang kebutuhan fungsional dari implementasi *node-node* tersebut. Pengalamatan IPv6 dan *short address* disetiap *node* sebagai kebutuhan komunikasi. Di dalam area jaringan 6LoWPAN, diidentifikasikan untuk protokol *mac address node* yang lebih rendah ke dalam IEEE 802.15.4. Modifikasi 6LoWPAN *protocol stack* pada penjelasan bab sebelumnya, merupakan penyesuaian dari kebutuhan perancangan penelitian. Dapat dilihat pada tabel dibawah ini.

Tabel 5.6 Fungsional Protokol Stack 6LoWPAN

Layer	Implementasi	Fungsional
Application	CoAP (*Contrained Application Protocol*)	*Web transfer protocol* khusus untuk penggunaan *node* dan jaringan terbatas dengan model interaksi *request/respons* seperti *HTTP* namun beroperasi pada UDP.
Transport	UDP (*User Datagram Protocol*)	*Lightweight* protokol untuk menghemat sumber daya memori dan prosesor. *Connectionless* yang dapat digantikan dengan *Confirmable request/response* untuk pertukaran pesan.
Network	IPv6 (*Internet Protocol version 6*)	Pengurangan beban penggunaan *ipv4* bagi mesin yang membutuhkan pengalamatan dalam infrastruktur jaringan yang ada.
Adaptation	6LoWPAN (*Ipv6 over Low power Wireless Personal Area Network*)	Jenis jaringan nirkabel untuk perangkat interkoneksi disekitar area kerja tertentu dengan penekanan mis. 60 *byte header* menjadi 7 *byte* dan mengoptimalkan mekanisme untuk jaringan nirkabel.
Physical and Data Link	*IEEE 802.15.4*	Teknologi radio berdaya rendah yang berkomunikasi satu sama lain dengan frekuensi 2,4 *GHz* tidak berlisensi dan kompatibel dengan IC dan ETSI.

Pengembangan untuk jaringan 6LoWPAN sudah banyak dilakukan. Penggunaan *library linux wpan* yang tersedia di *github* sebagai rujukan. Pengembangan dalam penelitian ini dilakukan *cross compiling kernel*. Agar sistem operasi didalam raspberrypi yang digunakan dapat mendukung jaringan 6LoWPAN tersebut. Untuk kebutuhan implementasi *linux wpan* akan dijelaskan lebih detail pada sub bab implementasi.



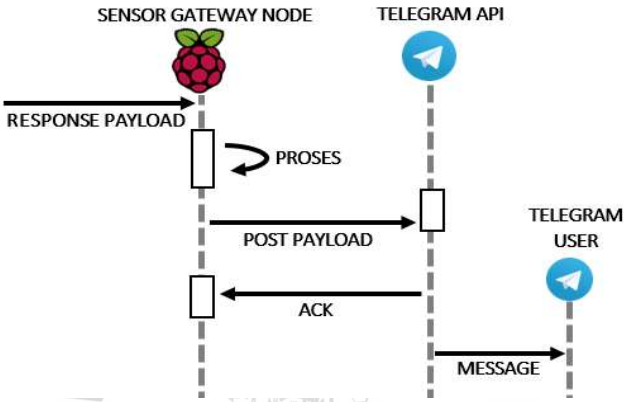
Gambar 5.7 Sequence Diagram Layer Protokol CoAP

Dalam pola *request/response message* yang diterapkan dalam protokol CoAP memiliki *payload code*. *Payload code* yang berupa `CONTENT` untuk memvalidasi *request* dari sensor *gateway*. Untuk mengimplementasikan protokol CoAP harus disediakan *resource* tersendiri untuk protokol tersebut. Berbeda dengan protokol MQTT dengan pola *publish/subscribe* yang memiliki satu topik. Menggunakan salah satu library yang telah dikembangan, penulis mengusulkan untuk menerapkan metode tersebut yang menyediakan *resource* untuk setiap *node*. Sebagai contoh, apabila *sensor-node* diterapkan sebuah *resource* kualitas air maka skema yang sama juga perlu diterapkan pada *gateway* sensor.

Pada Gambar 5.7 merupakan proses bagaimana kedua *node* bisa saling memvalidasi *request/response*. Di dalam *sensor-node* terdapat sensor untuk kualitas air. Ketika *node* sudah diaktifkan, maka akan dalam status *standby* dan menunggu *request.* Namun dalam keaadan tersebut, *sensor-node* sudah aktif membaca sensor hanya saja tidak ditampilkan. Ketika sensor *gateway* mulai mengaktifkan perintah *request message*, *sensor-node* melakukan validasi terhadap *payload code request* tersebut. Baru setelah itu ketika sudah tervalidasi, akan langsung mengirimkan *response payload*. Yang berupa nilai hasil baca sensor kualitas air. Data sensor yang diterima bisa langsung ditampilkan pada *terminal* Linux yang terdapat di *gateway* sensor. Namun penelitian ini, hasil sensor tersebut langsung dikirimkan menuju API Telegram. Agar dapat terhubungan dengan *chatbot* yang menampilkan hasil baca sensor kepada pengguna.

5.2.5 Perancangan Application Gateway

Penelitian ini menerapkan pengguna aplikasi berbasis API Telegram. Menggunakan *chatbot* yang bisa mengirimkan pesan secara otomatis. Untuk bisa menampilkan hasil pemantauan kualitas air. Perancangan ini menggunakan *method* sendMessage menggunakan skema pengiriman POST. Endpoint untuk perancangan ini adalah pengguna aplikasi Telegram yang menambahkan aku *chatbot* tersebut. Protipe yang dikembangan menggunakan kebutuhan khusus seperti penggunaan json sebagai tipe data konten.

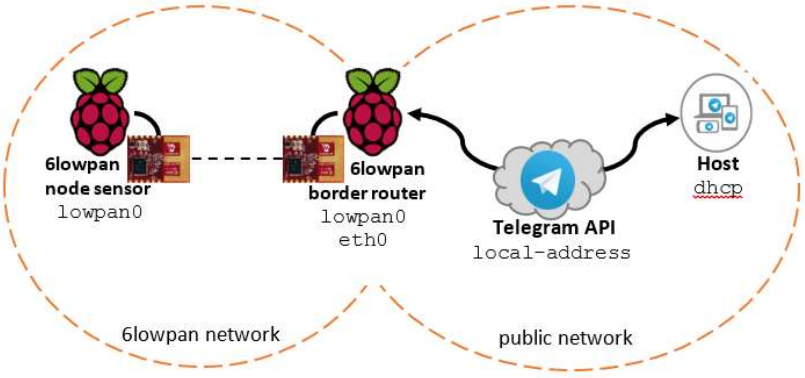


Gambar 5.8 Sequence Diagram Telegram Chatbot

Pesan yang dikirmkan oleh API Telegram sebagai message kepada pengguna, berisikan tampilan hasil kualitas air beserta *code* status. Fitur notifikasi pada *chatbot* dinonaktifkan. Tetapi ketika mendeteksi warning pada *code* status, otomatis akan mengaktifkan notifikasi.

5.2.6 Perancangan Topologi Jaringan

Agar sistem berjalan sesuai dengan yang diharapkan, perlu dibuat rancangan topologi jaringan antar komponen dan jaringan dalam sistem. Topologi jaringan ini mencerminkan bagaimana *sensor-node* dapat terhubung dan mengirimkan data ke *node border router*. Sekaligus menampilkan data tersebut ke aplikasi. Rancangan topologi ini dijelaskan pada Gambar 5.9 di bawah ini.



Gambar 5.9 Topologi Jaringan

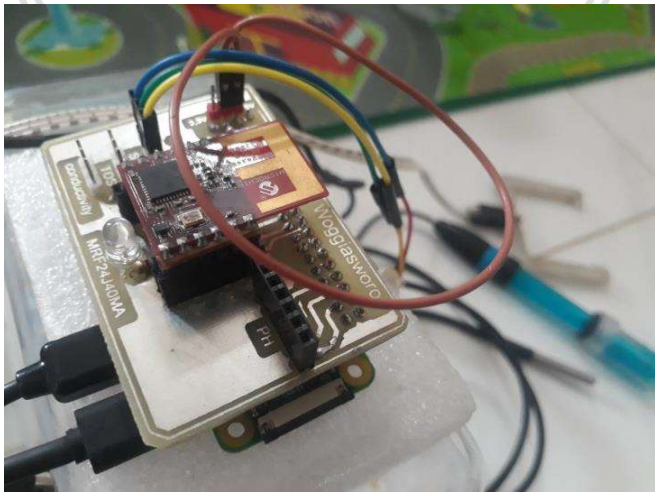
Raspberry Pi B+ yang berperan sebagai *node border router*. Dan berperan untuk terhubung melalui `eth0` ke *access point*. Untuk menunjang hal ini dibutuhkan juga perangkat keras lainnya yang menunjang dalam penelitian ini adalah *access point* TP-LINK. Bertindak sebagai *wireless adapter* untuk *user* dapat mengakses hasil data monitoring. Terdapat dua buah *interface* yaitu `lowpan0` untuk komunikasi antar *node* dan `eth0`. Dikonfigurasikan untuk menjalankan *ip dhcp* pada TP-LINK. Ketika *node border router* dijalankan, RaspberryPi memerlukan port 5683 untuk CoAP. *Sensor-node* akan berkomunikasi dengan *node border router* melalui interface `lowpan0`.

5.3 Implementasi Sistem

Berdasarkan analisis kebutuhan dan perancangan yang telah dilakukan sebelumnya, sub bab ini menjelaskan tentang implementasi yang dilakukan dalam penelitian ini. Implementasi tersebut meliputi implementasi 6LoWPAN dan protokol komunikasi CoAP, yang terdiri dari implementasi *sensor-node* dan implementasi *node border router*. Implementasi *application gateway* sebagai penyedian *platform open-source* untuk kebutuhan interaksi dengan *end-user*. Juga dijelaskan implementasi sensor, aplikasi dan topologi jaringan untuk kebutuhan pengujian.

5.3.1 Implementasi Sensor Node

Implementasi *sensor-node* ini dilakukan sebagai node pengirim. Sesuai dengan perancangan yang telah dijelaskan, *sensor-node* bertugas sebagai sistem pemantauan kualitas air. Rangkaian pada sistem ini terdiri dari RaspberryPI Zero sebagai mini komputer atau mikrokontroler. Modul MRF24J40MA sebagai modul *wireless*. Sensor suhu DS18B20 sebagai indikator pemantauan nilai suhu di dalam air. Sensor PH sebagai penentuan kadar pH apakah normal, asam atau basa. Dan sensor photodioda sebagai indikator tingkat kekeruhan air. Di bawah ini terdapat Gambar 5.10 yang menjelaskan bagaimana *sensor-node* diimplementasikan.



Gambar 5.10 Implementasi Perangkat Keras Sensor Node

Tabel 5.7 Serial Communication Arduino

Program sensor_analog.ino

```
 1  void kenernihan_air();
 2  void Ph_air();
 3  #define sensorpinph A1
 4  #define ldr         A0
 5  ../
14  ../
15  void setup() {
16    // put your setup code here, to run once:
17   Serial.begin(9600);
18    pinMode(ldr, INPUT);
19    pinMode(sensorpinph, INPUT);
20  }
21  ../
```

Pembacaan nilai sensor untuk mendeteksi kualitas air dilakukan dengan dua cara. Karena terdiri dari sensor analog dan digital, yaitu sensor ph dan sensor kekeruhan yang merupakan sensor analog dan sensor suhu yang merupakan sensor digital. Penggunaan Arduino Nano sebagai ADC (Analog Digital Converter) berfungsi untuk membaca kedua sensor analog tersebut. Sedangkan sensor digital, langsung dihubungkan dengan pin GPIO RaspberryPi. Kemudian untuk sensor analog akan mengirimkan hasil baca melalui *serial comm* kepada RaspberryPi. Ketika RaspberryPi mendapatkan serial data maka langsung dilakukan penggabungan data. Karena baik sensor analog maupun digital melakukan pembacaan data sensor secara bersamaa.

Gambar 5.11 Sensor Kualitas Air yang digunakan

Pada implementasi perangkat lunak *sensor-node* menggunakan command line linux. Menggunakan bahasa pemrograman Python3 yang langsung menjalankan program tanpa melakukan *compiling code.* Terdapat satu program utama yang memanggil keseluruhan fungsi sistem dengan mencantumkan library di dalam

program utama tersebut. Program utama pada *sensor-node* berlatar belakang sebagai program protokol CoAP. Ditambahkan dengan fungsi dan parameter untuk melakukan pembacaan nilai sensor analog dan digital.

Tabel 5.8 Library dan Program Sensor

Program responseGETcoap.py

```
1  #!/usr/bin/env python3
2
3  import asyncio
4  import aiocoap.resource as resource
5  import aiocoap, time, datetime
6  import os
7  import serial
8  import re
9
10 def serialraspi():
11     ser = serial.Serial('/dev/ttyUSB0', 9600)
12     val = str(ser.readline())
13     return val
14
15 def sensorsuhu():
16     for i in os.listdir('/sys/bus/w1/devices'):
17         if i != 'w1_bus_master1':
18             ds18b20 = i
19     return ds18b20
20 ../
```

Penjelasan alur program dimulai melalui inisiasi main program. Proses berjalannya pengiriman pesan pada program ditentukan ketika *sensor-node* menerima *request code* dari *node border* router. Sedangkan untuk sinkronisasi, *sensor-node* akan menambahkan sebuah *resource* sebagai medianya. Sebelumnya program hanya menunggu *request code*, setelah itu baru memulai membaca alur yang terdapat didalam fungsi *resource* CoAP. *Request code* yang digunakan sebagai *payload* CoAP adalah *code* CONTENT. Dan terdapat didalam library aiocoap.resource. Dapat dilihat pada Tabel di bawah ini.

Tabel 5.9 Inisialisasi CoAP Resource

Program responseGETcoap.py

```
44 ../
45 class TemperatureResource(resource.Resource):
46   def __init__(self):
47     super(TemperatureResource, self).__init__()
48 ../
49 ../
58 def main():
59   root = resource.Site()
60
61   root.add_resource(('temperature',),
   TemperatureResource())
62   asyncio.async
   (aiocoap.Context.create_server_context(root))
63 ../
```

Setelah melakukan inisialisasi pesan confirmable terhadap *node border router,* program yang berjalan tetap berada dalam fungsi Resource tersebut. Karena didalamnya terdapat kelanjutan program ketika masing-masing *node* sudah saling tersinkronisasi. Kemudian akan melakukan pemanggilan fungsi pembacaan sensor. Yang sebelumnya telah diuraikan bahwa terdapat dua macam metode pembacaan sensor. Dengan menerapkan GET *resquest/response*, program akan memulai melakukan pembacaan pada fungsi ketika menerima GET *request* dari *node border router*. Didalam fungsi render_get terdapat pula variabel yang bertugas seperti melakukan *repackage* parameter terhadap pesan. Karena pesan-pesan berupa data sensor tersebut harus dijadikan dalam satu paket terlebih dahulu. Setelah itu baru bisa dimasukkan sebagai *payload* oleh protokol CoAP. Untuk lebih jelasnya program yang telah diimplementasikan dalam penelitian ini, dapat dilihat pada bagian lampiran program utama.

Tabel 5.10 Render GET CoAP

Program responseGETcoap.py

```
49  ../
50  def render_get(self, request):
51    sensor_ardun = serialraspi()
52    temp_sensor = sensorsuhu()
53    temp_sensor = str(read(temp_sensor))
54    payload = repack(temp_sensor,
    sensor_ardun).encode('utf-8')
55    return aiocoap.Message(code=aiocoap.CONTENT,
    payload=payload)
56  ../
```

Pada *sensor-node* juga diimplementasikan tentang penggunaan jaringan 6LoWPAN. Agar dapat terhubung melalui jaringan 6LoWPAN, masing-masing *node* melakukan konfigurasi untuk mengalamati perangkat. Perintah yang dilakukan berupa *linux command line* karena menyangkut terhadap konfigurasi di dalam sistem operasi. Tiap-tiap perintah bisa digabungkan menjadi satu perintah pemanggilan. Menggunakan Shell *programming* dengan ekstension file `.sh`. Program shell yang telah dibuat memiliki nama yang sama untuk masing-masing *node*. Yakni, `launchwpan.sh`. Program shell tersebut disimpan di direktori `/root` agar semua akses dapat berjalan dengan baik. Berikut ini akan mengurai beberapa perintah yang dilakukan.

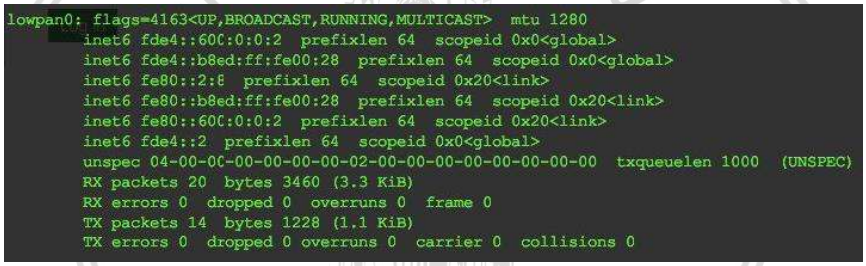
```
$ ip link set dev wpan0 address 04:0:0:0:0:0:0:2
$
$ iwpan dev wpan0 set pan_id 0xbaed
$ iwpan dev wpan0 set short_addr 0x028
$
$ ip link add link wpan0 name lowpan0 type lowpan
```

Perintah-perintah yang dilakukan pada tahap awal merupakan sebuah pengalamatan. Karena sebelumnya interface yang digunakan belum memiliki pengalamatan. Hanya telah dikenali sebagai perangkat komunikasi. Sehingga harus diberikan pengalamatan secara manual yang berbasis IPv6. Konfigurasi pada `wpan0` sebagai MAC Address. Sedangkan setting `pan_id` dan `short_addr`

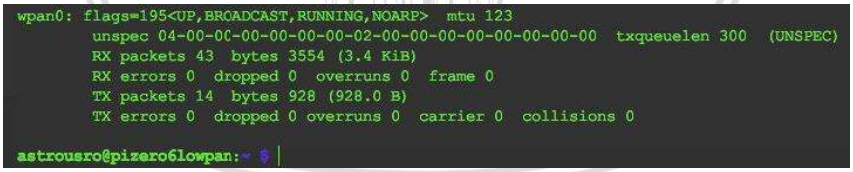
merupakan bagian dari `wpan0` itu sendiri. Lalu selanjutnya diidentifikasikan bahwa `wpan0` juga berlaku sebagai bagian `lowpan0` dengan tipe *interface* `lowpan`.

```
$ ip link set wpan0 up
$ ip link set lowpan0 up
$
$ ifconfig lowpan0 inet6 add fe80::2:8/64
$
$ ip addr add fde4::2/64 dev lowpan0
```

Tahapan kedua dalam pemberian alamat pada *sensor node* adalah pengalamantan IPv6 untuk *interface* `lowpan0`. Sebelum pengalamatan IPv6, `up link` terlebih dahulu dilakukan. Baik pada *interface* `wpan0` maupun `lowpan0`. Agar kedua interface tersebut benar-benar terbukti aktif. Alamat IPv6 yang diberikan merupakan pilihan acak atau menyesuaikan dengan kebutuhan implementasi. Dan berada dalam satu lingkup jaringan *local host*. Kemudian ditambahkan pula alamat IPv6 untuk kebutuhan *global route*. Sehingga saat proses melakukan komunikasi tidak perlu menspesifikan alamat interface. Alamat IPv6 untuk kebutuhan *global route* ditambahkan ke dalam *interface* `lowpan0`. Sehingga *interface* tersebut memiliki 2 alamat IPv6. Alamat IPv6 untuk kebutuhan local host dan untuk kebutuhan *global route*.



Gambar 5.12 Interface LOWPAN Sensor Node

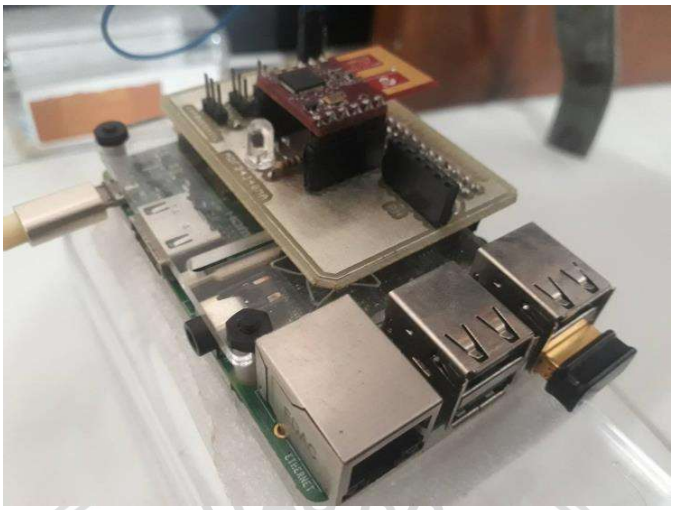


Gambar 5.13 Interface WPAN Sensor Node

5.3.2 Implementasi Node Border-Router

Implementasi *noder border-router* ini dilakukan sebagai node penerima atau bisa dikatakan sebagai *sensor gateway*. Seperti yang telah diuraikan pada bab perancangan, implementasi perangkat keras hanya meliputi Raspberry Pi Type B+ sebagai mini komputer atau mikrokontroler. Dan modul MRF24J40MA sebagai *wireless* modul. Terdapat penambahan perangkat keras berupa USB Wifi Dongle atau Kabel UTP untuk terhubung ke *access point*. Implementasi dalam penelitian ini lebih menggunakan USB Wifi Dongle agar dapat langsung terhubung ke jaringan internet. Hal ini diperlukan karena membutuhkan koneksi internet untuk menghubungkan dengan API Telegram. Konfigurasi yang dilakukan terhadap USB

Wifi Dongle dengan menambahkan keterangan SSID dan Password dari koneksi yang tersedia.



Gambar 5.14 Implementasi Perangkat Keras Node Border Router

Pada proses implementasi perangkat lunak dalam *node border router* ini, secara teknis sama seperti *sensor-node* dalam berperan sebagai pengirim. Karena ketika berperan sebagai pengirim, *node border router* menghubungi API Telegram. Untuk dapat meneruskan pesan yang diterima ke dalam aplikasi *chatbot*. Jadi, *node border router* memiliki dua peran yakni sebagai penerima pesan dan sebagai pengirim pesan. Layaknya sebuah *gateway* dalam suatu *network*.

Dengan mengimplementasikan program utama dengan basis pemrogram *python*, *node border router* juga menggunakan program utama yang berlatar belakang protokol CoAP. Menggunakan Python3 dengan *library* aiocoap. Hanya saja terdapat penambahan *library* yang berhubungan dengan API Telegram. Berperan sebagai yang menerima pesan menggunakan *request code* GET. Node *border router* dianggap sebagai *client* oleh *sensor-node*.

Tabel 5.11 Library Program Utama Node Border Router

Program requestGETcoap.py

```
 1  import asyncio
 2
 3  import aiocoap
 4  from aiocoap import *
 5  import time, datetime
 6  from pprint import pprint
 7
 8  from formatter import *
 9  from sender import *
10
11  now = datetime.datetime.now()
12  ../
```

Tabel 5.11 merupakan *library* dan fungsi yang digunakan pada program utama. File *library* `formatter` dan `sender` berperan dalam melakukan hubungan terhadap API Telegram. Namun proses tersebut tetap berada dibagian akhir setelah menerima pesan dari *sensor-node*. Dalam program tersebut hanya berisikan fungsi `main` CoAP yang melakukan proses inisialisasi pertama untuk sinkronisasi. Ketika mengirimkan *request code*, *sensor-node* melakukan pembuatan *resource*. Setelah itu didefinisikan sebagai `protocol` oleh program utama *node border router*. Memberikan *response code* sementara berupa pembuatan `client context. Menjadikannya terbukti bahwa request tersebut valid`. Dan *sensor-node* siap memberikan response message berupa payload data sensor.

Tabel 5.12 CoAP Request Message

Program requestGETcoap.py	
12	../
13	@asyncio.coroutine
14	def main():
15	protocol = yield from Context.create_client_context()
16	
17	request = Message(code=GET)
18	request.set_request_uri ('coap://[fde4::2]/temperature')
19	
20	try:
21	response = yield from protocol.request(request).response
22	../

Implementasi penggunaan jaringan 6LoWPAN menggunakan Shell *programming*. Sama seperti *sensor-node.* Langkah perintah konfigurasi yang dilakukan pun juga sama. Hanya pada pengalamatan *interface* yang berbeda. Namun sesuai dengan kebutuhan jaringan 6LoWPAN. Perbedaan pengalamatan `wpan0` hanya pada bit paling akhir untuk pengalamatan khusus sistem. Sedangkan untuk alamat `pan_id` sama seperti *sensor-node*. Dan untuk penglamatan `short_addr` dilakukan pengalamatan tersendiri, karena mewakili masing-masing perangkat.

```
$ ip link set dev wpan0 address 04:0:0:0:0:0:0:1
$
$ iwpan dev wpan0 set pan_id 0xbaed
$ iwpan dev wpan0 set short_addr 0x019
$
$ ip link add link wpan0 name lowpan0 type lowpan
```

Pemberian alamat *interface* lowpan0 melakukan konfigurasi yang sama dengan yang dilakukan pada konfigurasi *lowpan address sensor-node*. Ditambahkan pengaturan `gateway addr` untuk jaringan 6LoWPAN. Implementasi tersebut dilakukan agar sistem dapat menggunakan ULA IPv6 *global route*.

```
$ ip link set wpan0 up
$ ip link set lowpan0 up
$
$ ifconfig lowpan0 inet6 add fe80::2:7/64
$
$ set gateway addr 6lowpan side
$ ip addr add fde4::2/64 dev lowpan0
```

```
lowpan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1280
        inet6 fe80::600:0:0:1  prefixlen 64  scopeid 0x20<link>
        inet6 fe80::b8ed:ff:fe00:19  prefixlen 64  scopeid 0x20<link>
        inet6 fde4::1  prefixlen 64  scopeid 0x0<global>
        inet6 fe80::2:7  prefixlen 64  scopeid 0x20<link>
        unspec 04-00-00-00-00-00-00-01-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 1  bytes 72 (72.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 27  bytes 4136 (4.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Gambar 5.15 Interface LOWPAN Node Border Router

```
wpan0: flags=195<UP,BROADCAST,RUNNING,NOARP>  mtu 123
        unspec 04-00-00-00-00-00-00-01-00-00-00-00-00-00-00-00  txqueuelen 300  (UNSPEC)
        RX packets 3  bytes 126 (126.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 43  bytes 3941 (3.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

astrousro@raspi6lbr:~ $
```

Gambar 5.16 Interface WPAN Node Border Router

5.3.3 Implementasi 6LoWPAN dan CoAP

Dalam sub bab ini terdapat dua komponen yang akan dijelaskan untuk proses implementasi. Komponen pertama yaitu proses implementasi jaringan 6LoWPAN (IPv6 over Low Power Wireless Personal Area Network). Implementasi 6LoWPAN terdiri dari implementasi perangkat lunak dan perangkat keras. Kebutuhan perangkat lunak yang dimaksud yakni, proses instalasi 6LoWPAN. Sedangkan kebutuhan perangkat keras meliputi RaspberryPi sebagai mikroprosesor dan MRF24J40MA sebagai wireless modul. Dan komponen kedua yang akan dijelaskan yaitu proses implementasi protokol komunikasi CoAP (Constrained Application Protocol). Ada beberapa opsi untuk menginstal jaringan 6LoWPAN ke dalam RaspberryPi. Banyak diantaranya menjelaskan petunjuk implementasi dengan sangat detail. Implementasi yang dilakukan pada bab ini berlaku untuk kedua node, baik itu Sensor Node maupun Node Border Router.

Bersumber dari platform dokumentasi GitHub, dengan menggunakan library linux-rpi untuk mengaktifkan *drivers* IEEE 802.15.4. Sebelum itu, penulis harus menginstal sistem operasi Raspbian terlebih dahulu, agar mini komputer Raspberry Pi dapat dijalankan. Saat melakukan penelitian implementasi jaringan 6LoWPAN, belum tersedia sistem operasi Raspbian yang menyediakan IEEE 802.15.4 *drivers support*.

Namun untuk melakukan pengembangan pada tahun ini sudah tersedia dengan kernel linux versi terbaru. Oleh sebab itu, penulis melakukan *cross compiling* kernel Raspbian dengan kernel linux ubuntu. Setidaknya dibutuhkan kernel linux versi 4.7+ atau versi 4.8+ untuk bisa mengkonfigurasikan jaringan

6LoWPAN. Untuk melakukan *cross compiling* kernel, langkah yang dibutuhkan yaitu menjalankan perintah:

```
$ raspi-update
$
$ git clone –depth 1
  https://github.com/raspberrypi/linux.git
    --branch rpi-4.7.y --single-branch linux-rpi2
$
$ make bcm2709_defconfig
```

Perintah `make` tersebut dijalankan untuk membuat file konfigurasi BCM2709. Yang digunakan sebagai *image* untuk proses *compiling* kernel dengan *library* linux-rpi2. Melakukan proses git clone terharap alamat linux-rpi2 tersebut. Untuk mendapat semua file linux-rpi2 yang terdapat di dalam alamat git clone.

Setelah menjalankan perintah `make`, selanjutnya menjalankan perintah `menuconfig`. Masuk ke dalam menuconfig yang berisikan pengaturan kebutuhan kernel sistem. Seperti menu Networking Support, Device Drivers dan sebagainya. Berikut dibawah ini merupakan pilihan menu yang terlihat ketika menjalankan perintah `menuconfig`. Tampilan tersebut muncul di dalam Linux *command line interface*. Karena sistem operasi yang digunakan berbasis *text*.

```
Device Drivers
  --> Network device support
       --> IEEE 802.15.4 drivers
Networking Support
  --> Networking options
       --> IEEE Std 802.15.4 Low-Rate Wireless
           Personal Area Networks support
```

Proses diatas tersebut sangat berguna dan penting ketika melakukan *cross comping* kernel. Hal ini dikarenakan, pada tab menu Device Drivers dan Networking Support terdapat pilihan menu untuk mengaktifkan IEEE 802.15.4. Ketika telah selesai dalam proses `menuconfig`, sistem operasi akan melakukan configurasi terhadap perubahan yang dilakukan dalam `menuconfig`. Kemudian melakukan perintah terakhir sebelum sistem melakukan *compiling* kernel.

```
/* perintah melakukan cross compiling */
$ make zImage modules dtbs -j4

$ sudo make modules install dtbs install
```

Perintah *cross compiling* membuat sebuah *kernel image* baru untuk mengganti dengan kernel yang sebelumnya. Selain itu juga membuat file `modules` dan `dtbs` yang akan dilakukan instalasi setelahnya. Proses *compiling kernel* memakan waktu yang cukup lama. Jika menggunakan *cross compiling*, bisa membutuhkan waktu kurang lebih 5 jam. Sedangkan jika melakukan *compiling kernel* lansung menggunakan RaspberryPi bisa membutuhkan waktu yang lebih lama lagi. Karena *processor* yang dimiliki RaspberryPi tidak seperti *processor laptop*. Dengan menggunakan perintah `-j4` di dalam perintah `make zImage modules dtbs -`

`j4` yang berarti menggunakan jumlah *prosesor* untuk *compiling kernel* bisa lebih cepat.

Setelah *kernel images* berhasil dibuat, hal yang dilakukan selanjutnya yaitu melakukan install file `dtbs` dan `modules` yang telah didapatkan dari proses sebelumnya. Kemudian memindahkan file `zImages` ke dalam boot kernel agar bisa menggunakan file kernel yang akan telah diperbarui. Lalu menambahkan inisiasi ke dalam file konfigurasi yang terdapat di dalam direktori `/boot` sistem operasi. Perintah yang dilakukan yakni:

```
$ sudo cp arch/arm/boot/zImage /boot/kernel.4.7.2.img
$
$ sudo vim /boot/config.txt
$ kernel = kernel.4.14.4+.img
```

Setelah semua selesai, penulis menginisiasikan kernel versi 4.7+ sebagai kernel.4.14.4+. Langkah terakhir untuk memastikan apakah proses *compiling* berhasil adalah mengecek *kernel version* dari terminal di RaspberryPI.

Gambar 5.17 Versi Kernel Hasil Compiling

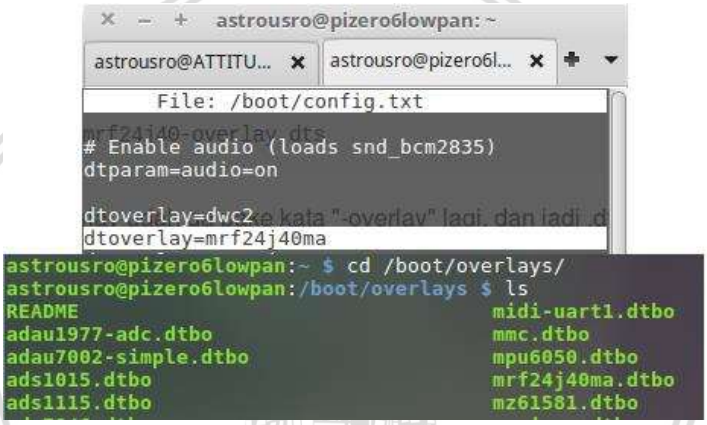
Langkah selanjutnya setelah selesai melakukan *compiling* kernel adalah konfigurasi 6LoWPAN dan menghubungkan wireless modul MRF24J40MA. Langkah pertama harus menginstal beberapa *package* untuk sistem operasi. Seperti `libnl`, `libnl-genl` sebagai pendukung dalam pengguna Bahasa pemrograman Python3 dan package `dh-autoreconf` untuk mengenali port MRF24J40MA dengan SPI Pin RaspberryPi. Kemudian setelah berhasil melakukan instalasi *package* pendukung, membuat program untuk MRF24J40MA sebagai `dtoverlay`. File ekstension yang dibuat menggunakan format `.dts` agar menghasilkan file format `.dtbo`. Perintah yang dijalankan yakni:

```
$ sudo apt-get install libnl-3-dev libnl-genl-3-dev
$ sudo apt-get install dh-autoreconf
$
$ dtc -@ -O dtb -o mrf24j40.dtbo mrf24j40-overlay.dts
```

Setelah berhasil menjalankan beberapa perintah tersebut, langkah selanjutnya adalah memindahkan file `.dtbo` yang diperoleh ke dalam folder `/boot/overlays`. Serta menambahkan pada baris paling bawah `/boot/config.txt` dengan perintah `dtoverlay = mrf24j40ma`. Untuk lebih jelasnya dapat dilihat pada Tabel 5.13 di bawah ini dan penjabaran *pseudocode* program *overlay* MRF24J40MA.

Tabel 5.13 Pseudocode MRF24J40MA Overlay

pseudocode program MRF24J40MA	
1	SET compatible = bcrm, bcrm2708, bcrm2835
2	IF target = spi address
3	SET address-cell = 1, size = 0
4	IF init.mrf24j40 = mrf24j40, mrf24j40ma
5	SET registry 0
6	DO interrupst pin 23 and 8 to GPIO pin
7	ELSE spidev0
8	SET disable
9	ELSE spidev1
10	SET disable
11	END
12	END
13	END



Gambar 5.18 Proses Implementasi MRF24J40MA

Implementasi tentang 6LoWPAN termasuk ke dalam penelitian `linux-wpan`. Library linux-wpan memiliki beberapa *tools* yang telah dikembangan. Kesatuan tools tersebut berada dalam satu media *open-source*. Github merupakan sumber literasi yang banyak menyediakan library. Hal ini senada dengan penelitian ini. Langkah yang dilakukan adalah dengan *git clone* untuk library wpan-tools ke dalam RaspberryPi. Setelah berhasil melakukan *git clone* terhadap wpan-tools, pindahkan file tersebut ke dalam directori `/opt/src/`. Hal ini bertujuan agar konfigurasi untuk inisiasi pembangunan sistem dapat berjalan dengan baik.

Gambar 5.19 Inisialisasi Konfigurasi Sistem

Untuk melakukan konfigurasi wpan-tools harus berada di dalam direktori `/opt/src/wpan-tools`. Setelah berada didalam direktori wpan-tools, melakukan konfigurasi otomotis menggunakan file yang telah disediakan dalam bentuk *shell programming*. Kemudian melakukan repackage kembali dengan

menggunakam perintah `make`. Baru setelah itu melakukan instalasi dengan perintah `make install` di dalam direktori wpan-tools. Perintah-perintah yang mencakup dalam penjabaran sebelumnya akan ditampilkan secara berurutan.

```
$ cd /opt/src/
$ git clone --depth 1 https://github.com/linux-wpan/
  wpan-tools.git wpan-tools
$ cd /opt/src/wpan-tools
$ ./autogen.sh
$ ./configure CFLAGS='-g -O0' --prefix=/usr
  --sysconfdir=/etc --libdir=/usr/lib make
```

```
$ make
$ sudo make install
```

```
config.status: executing libtool commands

        wpan-tools 0.7
        =====
fdir=/etc --libdir=/usr/lib
        prefix:                 /usr
        sysconfdir:             /etc

        compiler:               gcc
        cflags:                 -g -O0
        ldflags:

astrousro@pizero6lowpan:/opt/src/wpan-tools $
```

Gambar 5.20 Implementasi WPAN-Tools

Lahkah terakhir untuk implementasi 6LoWPAN yaitu melakukan *load module* MRF24J40MA. Agar dapat melakukan *load drivers* secara otomatis ketika mendeteksi perangkat keras MRF24J40MA. Perintah yang dilakukan yakni:

```
$ sudo modprobe mrf24j40
$ lsmod
```

Pada Gambar 5.21 di bawah ini merupakan hasil perintah yang dilakukan. Ketika dijalankan perintah lsmod, akan tampil semua keterangan tentang driver yang terdapat pada RaspberryPi. Dapat dilihat bahwa sebelumnya *interface* MRF24J40MA yang dalam gambar sebagai mrf24j40 belum mendeteksi perangkat. Setelah mrf24j40 bertugas sebagai yang menggunakan modul mac802154 yang tidak lain adalah IEEE 802.15.4.

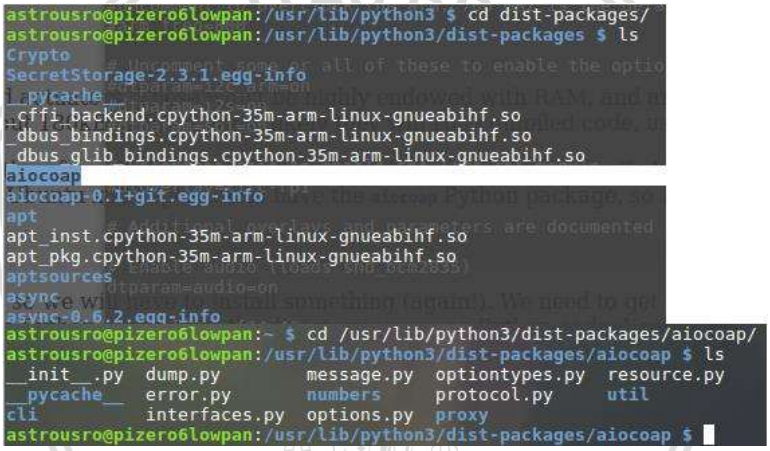
```
astrousro@pizero6lowpan:~ $ sudo modprobe mrf24j40
astrousro@pizero6lowpan:~ $ lsmod
Module                  Size  Used by
mrf24j40               13036  0
mac802154              60246  1 mrf24j40
ieee802154             77844  1 mac802154
crc_ccitt               1654  1 mac802154
snd_bcm2835            23054  0
snd_pcm                89110  1 snd_bcm2835
snd_timer              22332  1 snd_pcm
snd                    59220  3 snd_timer,snd_bcm2835,snd_pcm
udc_core               38875  5 usb_f_ecm,dwc2,u_ether,usb_f_rndis,
ip_tables              12363  0
x_tables               21353  1 ip_tables
ipv6                  394512  16
astrousro@pizero6lowpan:~ $
```

Gambar 5.21 Mengaktifkan Hardware MRF24J40MA

Seteleh menyelesaikan kebutuhan untuk implementasi jaringan 6LoWPAN, hal berikutnya adalah protokol CoAP. Sesuai dengan spesifikasi kebutuhan antarmuka perangkat lunak bahwa implementasi protokol CoAP menggunakan *library* `aiocoap` dan Bahasa pemrograman Python3. Pemilihan bahasa pemrograman tersebut karena *library* `aiocoap` hanya berjalan dalam Python3. Langkah pertama yang dilakukan yaitu menginstal `pytho3-aiocoap` untuk bisa menggunakan python asynchronous.

```
$ sudo apt-get install python3-aiocoap
$
$ git clone --depth=1
  https://github.com/chrysn/aiocoap.git
$ cd aiocoap/
$ sudo mv aiocoap /usr/lib/python3/dist-packages
```

Perintah-perintah diatas merupakan urutan perintah yang dilakukan untuk instal aiocoap Python3 dan *git clone library* aiocoap yang ada di Github. Gambar 5.21 di bawah ini menunjukkan bahwa *library* tersebut telah ada di dalam RaspberryPi.



Gambar 5.22 Keterangan Library aiocoap

5.3.4 Implementasi Application Gateway

Implementasi menggunakan chatbot Telegram didasarkan dengan rancangan yang ada dalam bab sebelumnya. Yakni bab perancangan. Inisiasi yang dilakukan dalam implementasinya akan dijelaskan dengan *source* code pada Tabel 5.14. Program untuk melakukan koneksi dengan API Telegram berada didalam *node border router*. Karena sejatinya *node border router* yang memiliki tugas untuk meneruskan pesan yang diterima. Menggunakan bantuan dari API tersebut, pesan yang didapat oleh *node border router* yang berupa bytes, dikonversikan terlebih dahulu sebelum akhirnya diubah ke dalam format json.

Tabel 5.14 Inisialisasi Penggunaan API Telegram

Program sender.py

```
1  import json
2  import requests
3
4  url = "https://api.telegram.org/bot"
5  token =
   "638062928:AAEE6GOgU6NTcqc4TaVYxFB7beEGpA1xCpE"
6
7  method = "/sendMessage"
8  #chat_id = '339070753,331992971'
9  endpoint = url + token + method
10
11 def send(payload):
12 ../
```

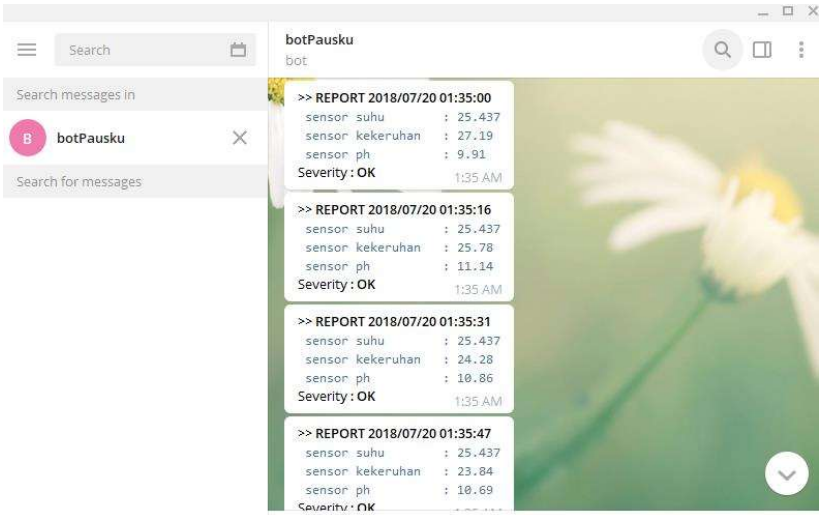
Singkronisasi antara API Telegram dengan chatbot, menggunakan parameter `chat_id`. Parameter tersebut didapatkan dari fitur chatbot untuk melihat parameter yang tersedia pada suatu akun. Didalam program *chatbot*, ditambahkan fungsi untuk menandakan status kualitas air. Range nilai yang diterapkan merupakan hasil dari standarisasi kualitas air oleh Kementrian Kesehatan. Dengan parameter yang digunakan adalah nilai suhu, kadar *pH* dan tingkat kekeruhan. Penjelasan program tentang pengaturan status kualitas air terdapat dalam Tabel 5.15 di bawah ini.

Tabel 5.15 Setup Status Kondisi Kualitas Air

Program formatter.py

```
3  ../
4  def message_formater(message):
5  ../
9  ../
10 sensor_name_1 = ' sensor suhu        : ' + matches[0]
   + '\n'
11     sensor_name_2 = ' sensor kekeruhan    : ' +
   matches[1] + '\n'
12     sensor_name_3 = ' sensor ph          : ' +
   matches[2]
13     severity = 'Severity : ' + get_severity(matches)
14 ../
17 ../
18 # Compare sensor value return severity
19 def get_severity(sensors):
20     dec_sensor_values = [Decimal(x.strip(' "')) for x
   in sensors]
21     if ((dec_sensor_values[0] > 27) or
   (dec_sensor_values[1] > 75) or (dec_sensor_values[2]
   < 5)):
22 return '*WARNING*'
23     else :
24         return '*OK*'
25 ../
```

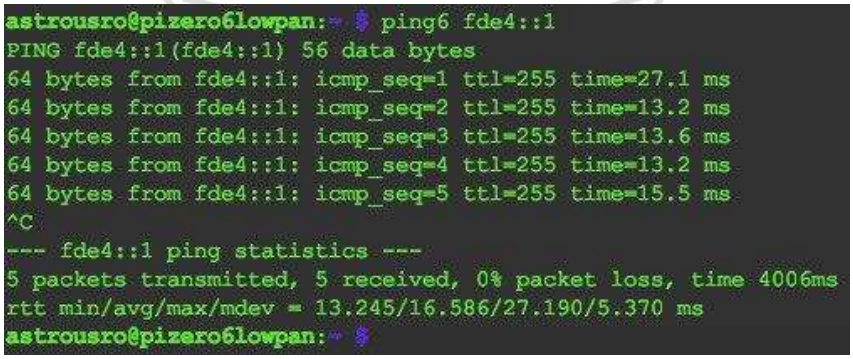
Berikut di bawah ini merupakan gambar hasil dari implementasi chatbot yang diterapakan. Chatbot ini hanya memiliki satu fungsi yaitu sebagai pengirim hasil pemantauan air. Dikarenakan hanya prototipe, maka implementasi dan perancangannya pun di buat dengan sangat sederhana. Hasil status kualitas air juga sudah langsung dapat dilihat beserta keterangan waktu penerimaan data monitoring kualitas air.



Gambar 5.23 Hasil Tampilan Chatbot Telegram

5.3.5 Implementasi Topologi Jaringan

Proses implementasi topologi jaringan membahas tentang konfigurasi yang dibutuhkan oleh *sensor-node, node border router, application gateway* dan pengguna. Ruang lingkup dalam hal *network area,* implementasi penelitian ini menggunakan koneksi rumah kontrakan. Jaringan tersebut merupakan salah satu provider terbesar di dalam negeri ini. Bermodalkan *access point*, implementasi topologi jaringan yang dirancang pada bab sebelumnya dapat terlaksana. Penerapan jaringan 6LoWPAN maupun jaringan *atau internet* dapat sangat baik digunakan ketika telah berada dalam satu lingkup area jaringan yang sama.



Gambar 5.24 Ping Menuju Node Border Router

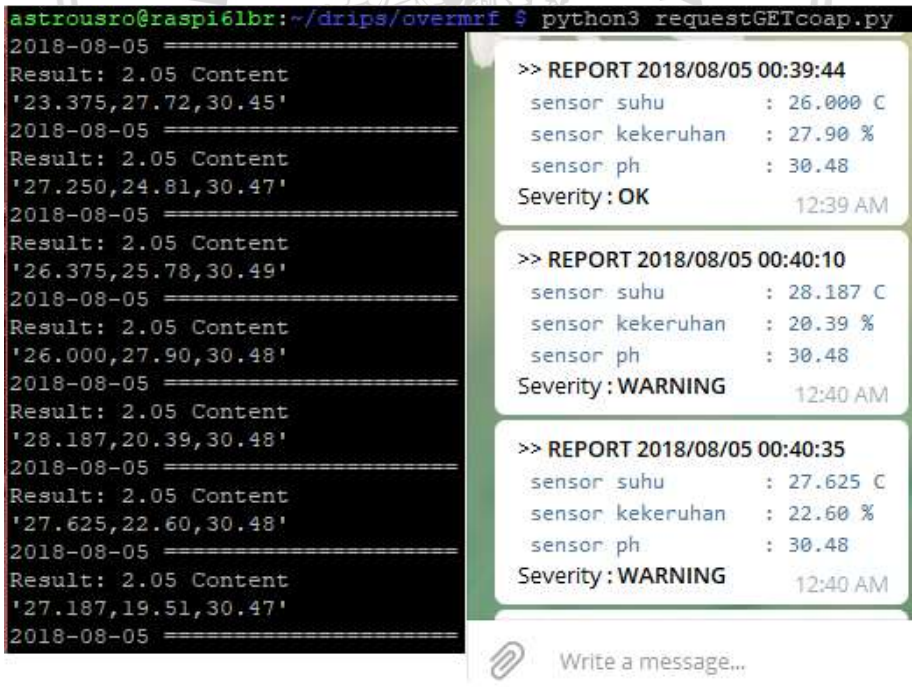
Konfigurasi untuk jaringan 6LoWPAN telah dijelaskan pada bab implementasi *sensor-node* dan implementasi *node border router*. Dengan menjalankan perintah yang terangkum di dalam *shell programming,* sudah dapat mengaktifkan jaringan

6LoWPAN. Masing-masing *node* memilki file `launchwpan.sh` sebagai pengalamatan untuk kebutuhan jaringan 6LoWPAN. Berikut akan diuraikan hasil implementasi yang dilakukan, baik itu dari *sensor-node* maupun *node border router*.

```
astrousro@raspi6lbr:~ $ ping6 fde4::2
PING fde4::2(fde4::2) 56 data bytes
64 bytes from fde4::2: icmp_seq=1 ttl=255 time=24.1 ms
64 bytes from fde4::2: icmp_seq=2 ttl=255 time=12.2 ms
64 bytes from fde4::2: icmp_seq=3 ttl=255 time=12.5 ms
64 bytes from fde4::2: icmp_seq=4 ttl=255 time=14.4 ms
64 bytes from fde4::2: icmp_seq=5 ttl=255 time=13.6 ms
^C
--- fde4::2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4006ms
rtt min/avg/max/mdev = 12.258/15.419/24.188/4.456 ms
astrousro@raspi6lbr:~ $
```

Gambar 5.25 Ping Menuju Sensor Node

Gambar 5.26 di bawah ini merupakan tampilan hasil dari penerimaan data sensor dari *Sensor-Node*. Tahap awal penerimaan pesan masih dalam bentuk digit angka yang merupakan perwakilan dari setiap sensor. Dapat dikatakan bahwa data tersebut masih berupa data mentah. Yang kemudian diolah oleh *Node Border Router* menjadi data informatif. Pengolahan tersebut dilakukan sebelum menampilkan hasil pada akun *chatbot* Telegram.



Gambar 5.26 Hasil Pesan Disetiap Lingkup Jaringan

BAB 6 PENGUJIAN DAN ANALISIS

Tahapan pengujian dari 6LoWPAN dan protokol CoAP yang diimplementasikan pada sistem terbagi menjadi dua komponen pengujian yakni pada sistem integrasi dan sistem kualitas air. Sistem integrasi digunakan untuk menguji kebutuhan fungsional sistem. Pengujian ini dilakukan dengan menguji interaksi antar *node* yang telah dirancang untuk menjalankan satu fungsi tertentu. Sedangkan pengujian sistem kualitas air dilakukan untuk pengujian tingkat akurasi sensor yang digunakan dan telah dijabarkan pada bab perancangan. Kemudian dari hasil pengujian tersebut dilakukan analisis terhadap implementasi yang diharapkan.

6.1 Pengujian Sensor Kualitas Air

Pengujian ini bertujuan untuk mengetahui tingkat nilai akurasi sensor yang digunakan. Pengujian ini menggunakan metode pembandingan nilai dengan hasil baca sensor. Dengan penggunaan 3 jenis sensor, yakni sensor suhu, sensor PH dan sensor kekeruhan. Maka setiap sensor dilakukan pengujian tingkat akurasi nilai sensor.

Gambar 6.1 Parameter Pengujian Kualitas Air

6.1.1 Pengujian Sensor Suhu

6.1.1.1 Tujuan

Sensor suhu yang digunakan adalah sensor suhu DS18B20. Sensor tersebut merupakan sensor digital. Pengujian yang dilakukan untuk mengetahui nilai tingkat akurasi sensor dengan nilai sensor yang sebenarnya. Agar dapat memvalidasi data sensor yang akan dikirimkan sebagai nilai hasil pemantauan kualitas air.

6.1.1.2 Prosedur

Pengujian ini menjalankan program untuk pembacaan hasil baca sensor suhu DS18B20 pada *sensor-node*. Kemudian termometer sebagai pembanding untuk

sensor sesungguhnya melakukan pembacaan suhu. Langkah pengujiaan yang dilakukan adalah

1. Menjalankan *sensor-node* untuk melakukan pembacaan sensor
2. Running program sensor suhu menggunakan python.

Tabel 6.1 Program Sensor DS18B20

Program responseGETcoap.py

```
../
def read(ds18b20):
  location = '/sys/bus/w1/devices/' + ds18b20 +
'/w1_slave'
  tfile = open(location)
  text = tfile.read()
  tfile.close()
  secondline = text.split("\n")[1]
  temperaturedata = secondline.split(" ")[9]
  temperature = float(temperaturedata[2:])
  celsius = temperature / 1000

  hasilcelsius = 'Current temperature: %0.3f C' %
celsius
  return hasilcelsius
../
```

3. Menunggu data selesai dijalankan untuk mengambil data percobaan.
4. Mengamati data yang dihasilkan dan membandingkan dengan termometer asli.

6.1.1.3 Hasil

Hasil dari langkah pengujian dapat dilihat pada Tabel 6.2. Nilai pembacaan oleh sensor suhu terdapat pada *running program* Python di terminal linux. Satuan nilai yang digunakan untuk parameter suhu air adalah dengan satuan celsius. Sedangkan hasil suhu pada termometer dapat dilihat di dalam tabel beserta hasil nilai akurasi.

```
astrousro@pizero6lowpan:~/drips/sensor $ python3 temp_sensor.py
Current temperature : 24.312 C
Current temperature : 75.762 F
Current temperature : 24.312 C
Current temperature : 75.762 F
Current temperature : 24.312 C
Current temperature : 75.762 F
Current temperature : 24.250 C
Current temperature : 75.650 F
```

Gambar 6.2 Hasil Baca Sensor Suhu DS18B20

Tabel 6.2 Pengamatan Hasil Ukur

Data sampel ke-	Hasil Baca Sensor DS18B20	Pengukuran menggunakan thermometer	Selisih	Error	Akurasi
1	24,31 °C	23 °C	1,31	5,4 %	94,6 %

2	25,06 °C	24 °C	1,06	4,7 %	95,3 %
3	24,87 °C	24 °C	0,87	3,5 %	96,5 %
4	25,12 °C	24 °C	1,12	4,5 %	95,5 %
5	24,43°C	25 °C	0,57	2,3 %	97,7 %
Rata-rata Akurasi					95,9 %

6.1.1.4 Analisis

Berdasarkan hasil pengamatan dari pengujian yang dilakukan didapatkan nilai rata-rata akurasi sensor 95,9%. Nilai tersebut didapat dari perhitungan pembagian hasil baca sensor dengan hasil baca termometer. Kemudian dikalikan 100 untuk mencari nilai prosentase. Nilai eror yang sebagai optional didapatkan rata-rata akurasi 4,08%. Disimpulkan bahwa dari hasil rata-rata akurasi, penggunaan sensor suhu DS18B20 sebagai salah satu parameter kualitas air dapat memberikan hasil pemantauan yang optimal. Hal tersebut dikarenakan rata-rata akurasi yang didapat adalah berada pada nilai diatas 90%.

6.1.2 Pengujian Sensor PH

6.1.2.1 Tujuan

Pengujian akurasi sensor pH bertujuan untuk mengetahui berapa nilai akurasi pada sensor pH dengan kondisi cairan pada tempat pengujian yang sudah diatur sesuai kebutuhan pengujian yaitu cairan dalam kondisi netral, asam, dan basa.

6.1.2.2 Prosedur

Untuk melakukan pengujian ini prosedur yang dilalui adalah sebagai berikut,

1. Mempersiapkan cairan *sample* yang akan diuji ke wadah yang berbeda;
2. Mempersiapkan alat-alat yang diperlukan untuk pengujian;
3. Mencampurkan cairan asam atau basa pada air *sample* dalam wadah sesuai dengan kondisi yang diinginkan;
4. Meletakkan sensor pH ke dalam air *sample*;
5. Mencatat data hasil baca sensor pH terhadap cairan *sample* yang terlihat pada serial monitor.

6.1.2.3 Hasil

Hasil pengujian untuk akurasi nilai sensor pH dapat dilihat pada Tabel 6.2 di bawah ini. Menggunakan pH Buffer Kits sebagai pembanding terhadap nilai sensor dengan kategori nilai yang diujikan adalah 10 sebagai nilai basa, 7 sebagai nilai netral, 4 dan 3 sebagai nilai asam.

Tabel 6.3 Hasil Pengujian Akurasi Sensor PH

No	Nilai Buffer	Hasil Sensor pH	Akurasi
1	10	10.73	93.19 %
2	7	7.67	91.26 %

3	4	4.39	91.11 %
4	3	3.30	90.90 %
	Rata-rata		91.61%

6.1.2.4 Analisis

Berdasarkan pengujian yang telah dilakukan, didapatkan hasil seperti pada Tabel 6.3 dimana nilai tersebut diperoleh berdasarkan nilai hasil pengujian sensor yang dibandingkan dengan nilai buffer cairan *sample* yang dimana cairan *sample* sudah memiliki nilai pH yang pasti. Dapat dilihat nilai akurasi pada sensor sebesar 91.61% yang menandakan bahwa sensor sudah sangat baik dalam mendeteksi tingkat pH dalam cairan *sample* yang berkondisi netral, asam, dan basa. Nilai akurasi diperoleh menggunakan rumus yang sudah diuji coba yaitu:

$$Nilai\ akurasi = \frac{nilai\ buffer}{nilai\ sensor\ Ph} X\ 100\%$$

6.1.3 Pengujian Sensor Kekeruhan

6.1.3.1 Tujuan

Pengujian akurasi sensor kekeruhan bertujuan untuk mengetahui berapa nilai pada sensor kekeruhan dengan kondisi air pada tempat pengujian yang sudah diatur sesuai kebutuhan yaitu dengan mencampur air dengan tanah. Pada sensor kekeruhan hasil yang terlihat pada serial monitor diperoleh menggunakan rumus yaitu:

$$Nilai\ Sensor = \frac{Tegangan}{1024}\ X\ Inputan$$

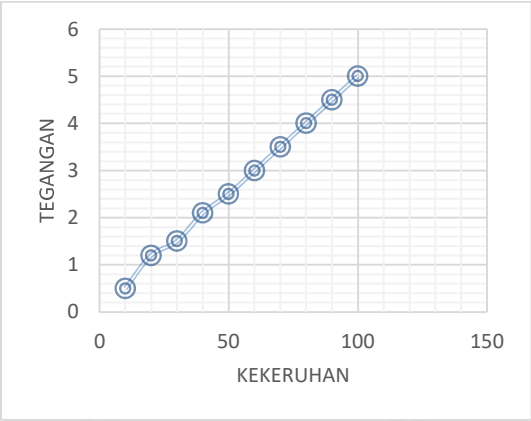
6.1.3.2 Prosedur

Prosedur melakukan pengujian:

1. Mempersiapkan air *sample* yang akan diuji ke wadah;
2. Mempersiapkan alat-alat yang diperlukan untuk pengujian;
3. Mencampurkan tanah pada air *sample* dalam wadah sesuai dengan kondisi yang diinginkan;
4. Meletakkan sensor kekeruhan ke dalam air *sample*;
5. Mencatat data hasil baca sensor kekeruhan terhadap air *sample* yang terlihat pada serial monitor.

6.1.3.3 Hasil

Hasil pengujian untuk nilai sensor kekeruhan dapat dilihat pada Gambar 6.3



Gambar 6.3 Grafik Sensor Kekeruhan

6.1.3.4 Analisis

Hasil sensor kekeruhan air yang terlihat pada serial monitor diperoleh menggunakan rumus yakni,

$$Nilai\ Sen \quad = \frac{Tegangan}{1024}\ X\ Inputan$$

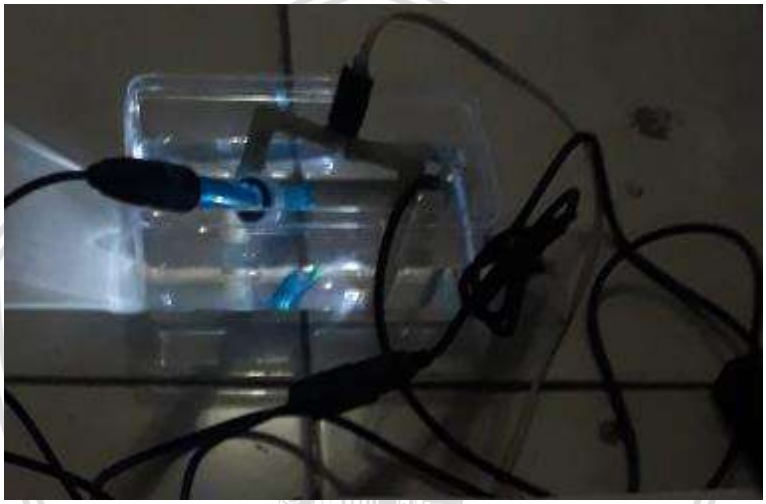
Rumus diatas dimaksudkan untuk mendapatkan nilai sensor kekeruhan dengan cara nilai tegangan sebesar 5V dibagi 1024 dikali *input* pada *range* 0 hingga 1024. Tergantung dari *input* photodiode. Maka didapatkan nilai hasil dari sensor dengan range 0-100. Nilai range ini diperoleh dari data kuisioner dalam penelitian tentang kekeruhan air menggunakan sensor kekeruhan berupa photodiode dan LED (Wiguna, 2018). Nilai *range* yang diterapkan adalah 0-20 untuk jernih dan 81-100 untuk keruh. Hasil perhitungan diatas adalah nilai tegangan yang dihasilkan dengan rentang nilai 0 hingga 5V dapat menghasilkan nilai pada sensor sesuai dengan yang diinginkan. Nilai pada sensor dapat dihasilkan dengan nilai 0-100 tergantung dari nilai *input* tegangan yang di terima oleh sensor photodioda. Disimpulkan bahwa dari hasil baca sensor kekeruhan dan akurasi rentang nilai yang didapatkan dari persebaran data kuisioner, maka penggunaan dari sensor ini sudah cukup baik sebagai parameter tingkat kekeruhan pada air.

6.2 Pengujian Sistem Integrasi

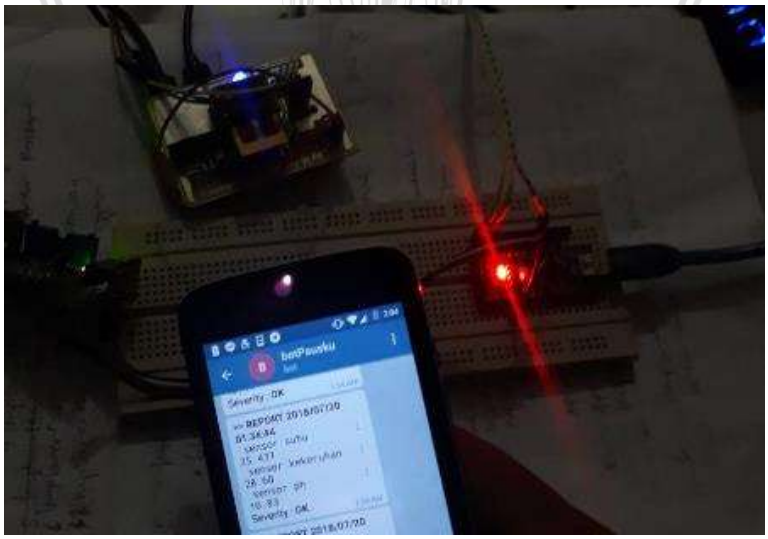
Pada pengujian ini dilakukan percobaan pengiriman data antar *node* untuk mendapatkan nilai waktu pengiriman dan kemampuan jangkauan jaringan 6LoWPAN. Parameter pengujian adalah *delay* dan *packet loss* berdasarkan jarak antar *node* yang digunakan sebagai pengujian terhadap *interface* jaringan 6LoWPAN. Untuk mengetahui kemampuan dari implementasi yang dilakukan berdasarkan perancangan perangkat keras. Ditinjau dengan menggunakan beberapa parameter *quality of service* dalam jaringan. *Delay* tersebut merupakan

waktu penerimaan data sensor yang dikirimkan *sensor*-node menuju *node border router.*

Hasil uji yang dilakukan akan ditampilkan dalam bentuk tabel. Implementasi *node border router* menerapkan sintaks berupa `print time.delay` dan keterangan zona waktu. Sintaks tersebut berfungsi untuk melakukan perintah menampilkan hasil waktu penerimaan data dalam satuan detik(s). *Delay* yang dijadikan parameter bukanlah hasil tersebut tersebut. Kalkulasi antara waktu pengiriman oleh *sensor-node* dengan waktu penerimaan oleh *node border router*. Hasil kalkulasi tersebut yang dijadikan sebagai nilai *delay* proses pengiriman data antar node. Sedangkan untuk parameter *packet loss* dibutuhkan untuk mengetahui kemampuan *node border router* agar dapat mengakses *resource* CoAP yang telah dibuat oleh *sensor-node* ketika program dijalankan.



Gambar 6.4 Perangkat Sensor Kualitas Air



Gambar 6.5 Hasil Output Data Sensor

6.2.1 Pengujian Performa Pengiriman Data

6.2.1.1 Tujuan

Dengan pengujian terhadap pengiriman data sensor ini diharapkan mendapatkan hasil kinerja dari sistem yang diterapkan dari segi komunikasi pengiriman data. Hal tersebut merupakan hasil dari implementasi jaringan 6LoWPAN dan protokol CoAP. Masing-masing dari setiap *node* harus saling terhubung dahulu melalui jaringan 6LoWPAN. Setelah itu, mengakses *resource* yang disediakan untuk protokol CoAP sebagai media dalam pertukaran pesan. Proses pengiriman pesan akan gagal ketika *sensor-node* gagal mendapatkan *resource* tersebut. Oleh karena itu, jangkauan jarak antar *node* sangat berpengaruh dalam proses awal komunikasi *node* tersebut.

6.2.1.2 Prosedur

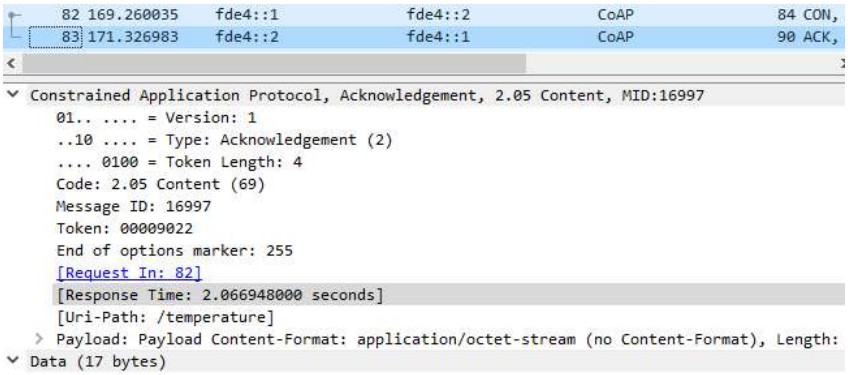
1. Mengaktifkan *sensor-node* sistem kualitas air menggunakan *powerbank* dan *node border router* menggunakan USB adaptor.
2. Menghubungkan USB WIFI Dongle dengan *sensor-node* melalui *port* USB RaspberryPI Zero sedangkan untuk *node border router* yang memiliki *port* LAN menggunakan kabel UTP untuk terhubung dengan Laptop.
3. Melakukan *remote access* (SSH) terhadap masing-masing *node* menggunakan Laptop dan aplikasi PUTTY dengan akses *hostname* dan *password* untuk tiap-tiap *node*.
4. Menjalankan program untuk mengaktifkan *interface* jaringan 6LoWPAN beserta alamat masing-masing *node*.
5. Menjalankan program `responseGETcoap.py` untuk *sensor-node* (*node* pengirim pesan) dan program `requestGETcoap.py` untuk *node border router* (*node* penerima pesan).
6. Mempersiapkan konfigurasi untuk pemantauan paket dalam proses komunikasi antar *node* menggunakan perintah `tcpdump` pada *node border router* sebagai penerima pesan sensor kualitas air.
7. Menjalankan aplikasi *wireshark* untuk menganalisis hasil *capture packet* yang dilakukan sebelumnya menggunakan perintah `tcpdump`.
8. Pengujian dilakukan pada ruangan bebas dengan jarak maksimal 50 meter dan proses pengambilan data selama 30 menit untuk kemudian dicari nilai rata-rata dari *response time* yang didapatkan sebagai parameter *delay*.

6.2.1.3 Hasil

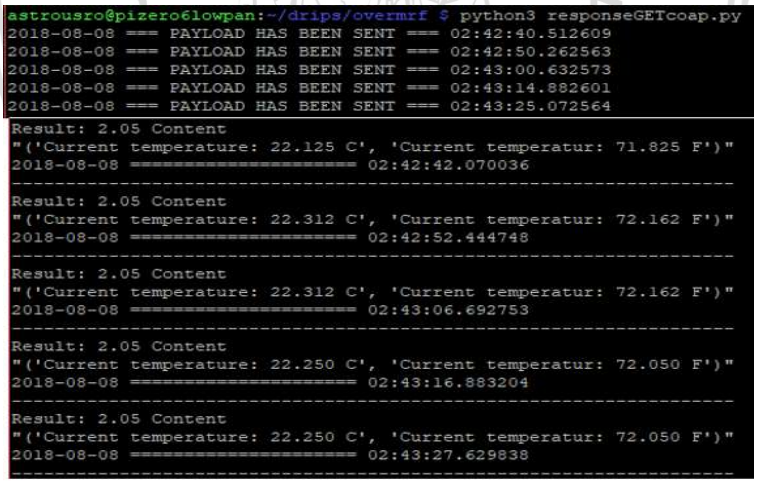
Berikut ini adalah hasil pengujian performa pengiriman data antar *node* dengan jarak maksimal yang telah ditentukan pada ruang bebas dan delay waktu pengiriman pesan tersebut. Hasil yang didapatkan menggunakan dua cara, yakni melalui terminal linux yang menampilkan keterangan waktu. Baik itu pada *sensor-*

node maupun *node border router*. Dan cara kedua melakukan *packet capturing*. Hasil pengujian dapat dilihat dalam Tabel 6.4.

Tabel hasil percobaan tersebut merupakan perolehan waktu *response time* yang diperoleh berdasarkan ketentuan jarak yang dilakukan ketika pengujian. Beberapa percobaan *request message* berguna untuk mendapatkan rata-rata nilai *delay* dari penerimaan pesan itu sendiri. Gambar 6.6 dan Gambar 6.7 dibawah ini merupakan proses dari perolehan nilai-nilai *response time* tersebut dari sisi *node border router*. Yang dimana proses tersebut terus berjalan ketika terjadi *request/response* oleh *node* untuk mendapatkan perolehan waktu dengan pengaturan jarak antar node.



Gambar 6.6 Capture Paket dengan Wireshark



Gambar 6.7 Tampilan Hasil Request/Response

Tabel 6.4 Hasil Pengujian Performa Pengiriman Antar Node

Percobaan Pengiriman	Jarak (m)	Waktu (s)	Keterangan
Percobaan ke-1	5	1,331	Berhasil Mendapatkan Resource
Percobaan ke-2	10	2,066	Berhasil Mendapatkan Resource
Percobaan ke-3	15	2,192	Berhasil Mendapatkan Resource

Percobaan ke-4	20	6,723	Berhasil Mendapatkan Resource
Percobaan ke-5	25	7,203	Berhasil Mendapatkan Resource
Percobaan ke-6	30	9,119	Berhasil Mendapatkan Resource
Percobaan ke-7	35	9,967	Berhasil Mendapatkan Resource
Percobaan ke-8	40	11,028	Berhasil Mendapatkan Resource
Percobaan ke-9	45	13,417	Berhasil Mendapatkan Resource
Percobaan ke-10	50	-	Gagal Mendapatkan Resource

6.2.1.4 Analisis

Berdasarkan pengujian yang dilakukan, keberhasilan proses pengiriman data sensor dipengaruhi dari berhasil atau tidaknya *node border router* yang melakukan *request* dalam memperoleh *resource*. Resource tersebut merupakan *resource* untuk komunikasi protokol CoAP yang berada pada *application layer* untuk masing-masing *node*. Pengujian tersebut dilakukan untuk mendapatkan cakupan jarak terjauh dari penggunaan *wireless module* MRF24J40MA sebagai radio frekuensi jaringan 6LoWPAN. Jarak terjauh yang diuji yakni 50 meter dengan status gagal mendapatkan *resource*. Oleh karena itu, penerapan jarak antar *node* sejauh 50m tidak dapat dilakukan. Namun dengan jarak 45 meter, *node border router* berhasil mendapatkan *resource* untuk protokol komunikasi CoAP. Dengan rata-rata waktu pengiriman adalah 13,417 detik. Dan jarak terdekat yang bisa diimplementasikan adalah 5 meter, dengan rata-rata waktu pengiriman sebesar 1,331 detik.

Hasil pengujian tersebut merupakan rentang waktu dari *node* menuju *node* lainnya. Belum termasuk waktu pengiriman ketika melalui API Telegram menuju *end-user*. Hal tersebut tidak dilakukan pengujian karena waktu penerimaan pesan kepada *end-user* dipengaruhi oleh penggunaan jaringan internet oleh *node border router* yang berperan sebagai *sensor gateway*. Dapat disimpulkan dari hasil pengujian performa pengiriman data antar *node* bahwa implementasi 6LoWPAN dan CoAP pada sistem *monitoring* kualitas air memiliki jarak jangkauan cukup jauh yakni 45 meter.

6.2.2 Pengujian Performa Interface LoWPAN

6.2.2.1 Tujuan

Pengujian ini bertujuan untuk mengetahui performa dari implementasi 6LoWPAN yang telah dilakukan. Mulai dari penggunaan *wireless module* MRF24J40MA dan konfigurasi 6LoWPAN yang dilakukan pada RaspberryPi sebagai *node* menggunakan sistem operasi linux. Pengujian ini didasarkan pada jaringan sensor nirkabel berbasis 6LoWPAN. Meskipun penerapan *routing protokol* dalam penelitian ini belum dilakukan, pengujian dapat dilakukan dengan parameter *delay* dan *packet loss* terhadap jarak antar *node* yang berada dalam lingkup jaringan 6LoWPAN. Kebutuhan pengujian ini juga sebagai acuan untuk performa

komunikasi antar *node* menggunakan protokol pertukaran pesan CoAP. Karena dalam protokol tersebut dibutuhkan koneksi awal untuk saling menginisialisasi antar *node*.

6.2.2.2 Prosedur

1. Mengaktifkan *sensor-node* sistem kualitas air menggunakan *powerbank* dan *node border router* menggunakan USB adaptor.
2. Melakukan *remote access* (SSH) terhadap masing-masing *node* menggunakan Laptop dan aplikasi PUTTY dengan akses *hostname* dan *password* untuk tiap-tiap *node*.
3. Menjalankan program untuk mengaktifkan *interface* jaringan 6LoWPAN beserta alamat masing-masing *node*.
4. Melakukan *ping interface* dari *node border router* menuju *sensor-node* untuk dilakukan pengujian dengan dua tipe ukuran paket yakni 56 bytes dan 128 bytes.
5. Pemilihan ukuran paket tersebut dikarenakan besaran paket yang diperoleh melalui jaringan 6LoWPAN dan protokol CoAP adalah 17 bytes berdasarkan hasil *capture wireshark*.
6. Menjalankan perintah `tcpdump` untuk menangkap hasil paket yang dikirimkan oleh *node* yang kemudian dilakukan analisis menggunakan aplikasi *wireshark*.
7. Pengaturan jarak antar *node* sama dengan pengujian yang dilakukan untuk mendapatkan performa pengiriman data antar yaitu pada ruangan bebas.

6.2.2.3 Hasil

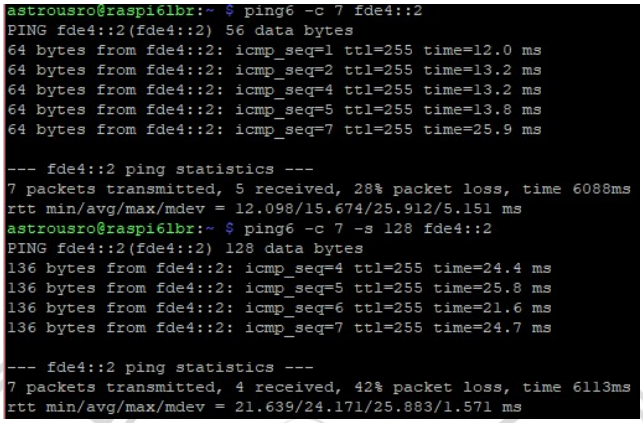
Berikut ini adalah hasil pengujian performa *interface lowpan* yang digunakan sebagai *network adapter* untuk komunikasi antar *node*. Pengujian ini dilakukan dengan melakukan *ping* terhadap *interface lowpan sensor-node* dengan ketentuan jarak dan ukuran paket. Masing-masing jarak dan ukuran paket memiliki hasil *capture* tersediri baik itu pada *interface lowpan* maupun *wpan*. Selanjutnya akan dianalisis menggunakan aplikasi *wireshark* untuk mendapatkan nilai *delay* dan *packet loss*. Hasil pengujian dapat dilihat dalam Tabel 6.5 di bawah ini. Hasil tersebut merupakan nilai rata-rata keselurahan dari *capture* paket yang dilakukan ketika melakukan pengujian.

Tabel 6.5 Hasil Pengujian Performa Interface LoWPAN

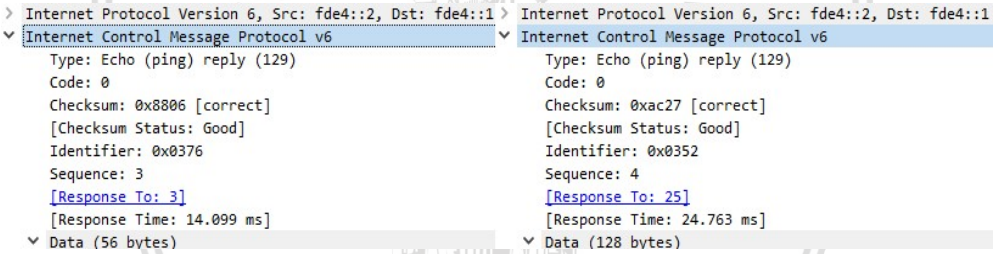
Jarak (m)	56 Bytes		128 Bytes	
	Delay (ms)	Packet loss rate	Delay (ms)	Packet loss rate
5	13.4	0 %	20.1	0 %
10	14.1	3 %	24.8	10 %
20	17.9	13 %	23.4	19 %

30	24.2	28 %	31.1	42 %
40	30.8	31 %	39.7	57 %

Berikut di bawah ini merupakan gambar hasil dari beberapa rangkaian proses pengujian yang dilakukan dalam mendapatkan rata-rata nilai untuk parameter *delay* dan *packet loss* terhadap jarak yang ditentukan. Masing-masing ukuran paket memiliki perintah dan hasil tersendiri ketika melakukan proses pengujian.

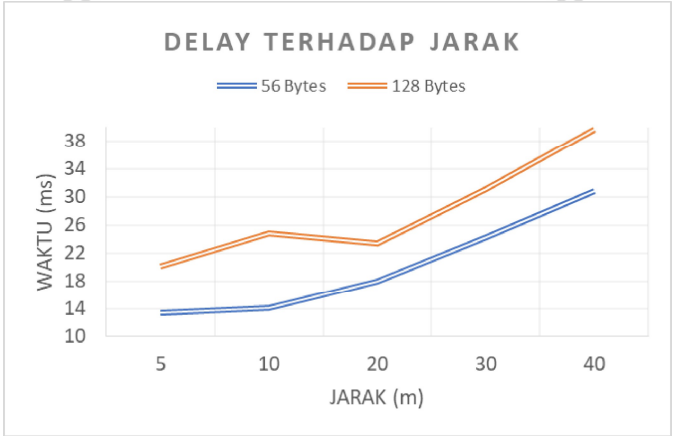


Gambar 6.8 Proses Pengujian Interface LoWPAN

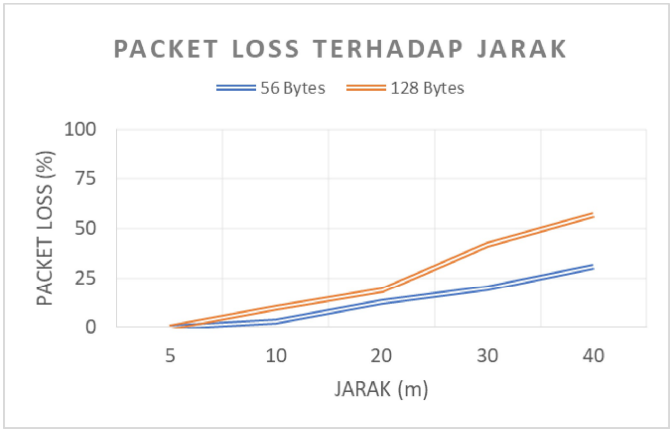


Gambar 6.9 Wireshark Capture Berdasarkan Ukuran Paket

Hasil rata-rata nilai pengujian dari masing-masing parameter ditampilkan dalam bentuk grafik. Agar dapat diketahui bagaimana performa *interface* 6LoWPAN yang digunakan berdasarkan penentuan jarak antar *node*.



Gambar 6.10 Hasil Rata-rata Delay terhadap Jarak



Gambar 6.11 Hasil Rata-rata Packet Loss terhadap Jarak

6.2.2.4 Analisis

Pengujian *interface* LoWPAN bertujuan untuk mengetahui performa dari jangkauan masing-masing *node* yang menggunakan MRF24J40MA sebagai *wireless* adapter. Kemampuan jangkauan *node* mempengaruhi kinerja sistem dalam proses pengiriman pesan melalui protokol CoAP. Hasil yang dimiliki berdasarkan parameter *delay* yakni sebesar 30.8 *ms* untuk ukuran paket 56 *bytes* dan 39.7 *ms* untuk ukuran paket 128 *bytes* dengan jarak antar *node* sejauh 40 meter. Berdasarkan hasil tersebut, performa implementasi jaringan 6LoWPAN masih dapat dikatakan baik. Nilai *delay* yang bertindak sebagai *response time* masih terbilang cukup ketika digabungkan dengan protokol CoAP. Proses jaringan 6LoWPAN itu sendiri melakukan *header compression* ketika terhubung dengan *resource* protokol CoAP yang menjadikan ukuran paket kurang lebih 17 *bytes*.

Sedangkan untuk parameter *packet loss*, dengan jarak 40 meter memiliki tingkat *packet loss* sebesar 31% untuk ukuran paket 56 *bytes* dan 57% untuk ukuran paket 128 *bytes*. Dengan jarak 40 meter yang menghasilkan *packet loss* diatas 50%, ini dapat mempengaruhi performa dalam pengiriman data. Namun jaringan 6LoWPAN merupakan lingkup jaringan terbatas yang memiliki tingkat kesalahan pengiriman yang cukup tinggi. Dapat disimpulkan, meskipun memiliki *packet loss rate* yang cukup tinggi dengan dukungan protokol komunikasi yang diimplementasikan yakni protokol CoAP performa pengiriman data akan memiliki hasil yang maksimal, seperti yang telah dijelaskan pada pengujian performa pengiriman data.

BAB 7 PENUTUP

Pada bab ini terdapat kesimpulan yang diambil berdasarkan tahapan-tahapan yang sudah dilakukan mulai dari perancangan arsitektur, implementasi, pengujian serta analisa data. Kesimpulan akan menjawab pertanyaan-pertanyaan pada rumusan masalah. Bab ini juga menampung saran-saran untuk penelitian selanjutnya.

7.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi dan pengujian yang telah dilakukan, didapatkan kesimpulan sebagai berikut :

1. Implementasi jaringan IPv6 pada sistem *monitoring* kualitas air dapat dilakukan dengan menerapkan arsitektur jaringan 6LoWPAN. Jaringan 6LoWPAN merupakan lingkup jaringan terbatas tidak stabil untuk perangkat *embedded* yang membutuhkan penggunaan daya yang rendah. Komunikasi 6LoWPAN berupa *sensor node* dan *sensor gateway*. Masing-masing memiliki alamat *IP address version 6* menggunakan *wireless* modul MRF24J40MA. *Wireless* modul tersebut menggunakan standar protokol jaringan IEEE 802.15.4 yang berimplikasi pada penggunaan daya yang rendah namun terbatas dalam lingkup jarak komunikasi antar node. Dengan menggunakan *interface* `lowpan0` dan `wpan0` sebagai pengalamatan. MRF24J40MA dapat terhubung satu sama lain. Konfigurasi alamat `lowpan0` pada *sensor gateway* sebagai *route gateway* agar interface `lowpan` dapat saling terhubung. Penggunaan Raspberry Pi pada setiap *node* merupakan salah satu kebutuhan implementasi 6LoWPAN dengan MRF24J40MA. Karena menggunakan Kernel Linux Versi 4.7+. *Node* sensor yang terdiri dari perangkat sensor suhu, sensor ph, dan sensor kekeruhan, tertanam dalam Raspberry Pi yang mengirimkan pesan melaui protokol CoAP menggunakan MRF24J40MA.
2. Mekanisme protokol CoAP yang diimplementasikan menggunakan request/response message. *Sensor-node* yang berperan sebagai pengirim *response message*, karena memiliki pesan data sensor kualitas air. Ketika program dijalankan, *sensor-node* membuat sebuah *resource* protokol CoAP menggunakan protokol dengan *message code* `CONTENT`. Selanjutnya *sensor gateway* berperan melakukan *request* dan memanggil *resource* yang disediakan *sensor-node* dengan alamat `lowpan0` yang didefinisikan untuk membuat protokol sebagai penerima dengan message code `Context`. Hal tersebut merupakan proses dari mekanisme *Message Confirmable* (CON). Yang menandakan bahwa *resource* untuk protokol CoAP telah siap digunakan sebagai protokol komunikasi antar node. Menggunakan *message code* dari *sensor gateway* berupa `GET` sedangkan *sensor-node* akan membalas dengan *payload* berupa `CONTENT`.
3. Kinerja implementasi jaringan IPv6 dan protokol CoAP sesuai dengan hasil dari pengujian sistem integrasi yaitu pengujian performa pengiriman data antar

node dan pengujian performa *interface* LoWPAN. Dengan menggunakan parameter pengujian *delay* dan *packet loss*, memiliki nilai rata-rata yang cukup mumpuni sebagai sistem monitoring kualitas air. Rata-rata nilai *delay* yang didapatkan yakni dengan jarak terpendek 5 meter adalah 13.4 *ms* untuk paket 56 *bytes* dan 20.1 *ms* untuk paket 128 *bytes*. Sedangkan untuk jarak terjauh 40 meter didapatkan rata-rata nilai *delay* 30.8 *ms* untuk paket 56 *bytes* dan 39.7 *ms* untuk paket 128 *bytes*. Tingkat keberhasil paket terkirim berdasarkan hasil rata-rata pengujian dengan paramater *packet loss* yaitu 100% untuk jarak 5 meter pada kedua tipe paket. *Packet loss rate* yang didapatkan pada jarak 5 meter adalah 0%. Sedangkan pada jarak terjauh 40 meter, *packet loss rate* untuk paket 56 *bytes* adalah 31% dan untuk paket 128 *bytes* adalah 57%. Sedangkan hasil pengujian performa pengiriman data antar *node* yang menggunakan protokol CoAP mendapatkan jarak terjauh 45 meter dengan nilai *delay* 13,417 detik dan jarak terpendek 5 meter dengan *delay* 1,331 detik. Dengan penerapan jaringan 6LoWPAN dan protokol CoAP terbukti dapat meminimalkan kesalahan pengiriman karena dengan menyediakan *resource* protokol sebagai inisialisasi awal komunikasi pertukaran data antar *node*.

7.2 Saran

Berdasarkan kesimpulan penelitian, maka penulis merekomendasikan berupa saran-saran untuk penelitian berikutnya yaitu:

1. Penelitian berikutnya dapat dilakukan untuk pengujian low power pada setiap node yang digunakan dan implementasi routing algoritma untuk jaringan IPv6.
2. Melakukan perbandingan protokol komunikasi seperti MQTT ataupun Websocket dalam implementasi jaringan 6LoWPAN dan menerapkan fitur keamanan untuk penggunaan protokol.
3. Pengujian protokol dapat dikembangkan, karena pada implementasi ini penulis belum menerapkan pengujian protokol CoAP. Misalnya pengujian untuk jenis ukuran payload.

DAFTAR PUSTAKA

Adarsh & Divya, B. D., 2014. Design of 6LoWPAN enabled Real Time Water Quality Monitoring System using CoAP. *APAN - Network Research Workshop,* 38(Proceedings of the Asia-Pacific Advanced Network), pp. 42-54.

Alfadoni, P., 2016. *Tunneling 6LoWPAN Protocol Stack in IPv6 Network.* ICST-2015 ed. Yogyakarta: Gajah Mada University.

Al-Fuqaha et al, A., 2015. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys & Tutorials,* 17(4), pp. 2347-2376.

Amani, F. & Prawiroredjo, K., 2016. Alat Ukur Kualitas Air Minum Dengan Parameter PH, Suhu, Tingkat Kekeruhan, dan Jumlah Padatan Terlarut. *JETri,* 14(kualitas air, Arduino, kekeruhan, pH), pp. 49-62.

Ankith, S., S, A., N, S. M. & Mallela, P., 2015. *Design of Ipv6 Network Enabled Smart Water Flow.* Kuala Lumpur, Malaysia, APAN.

Anwari, H., 2016. *Pengembangan IoT Middleware Sebagai Gateway Untuk Protokol CoAP, MQTT dan Websocket.* Malang: Fakultas Ilmu Komputer, Universitas Brawijaya.

BPPT, B. P. d. P. T. -., 2012. *Online Monitoring Kualitas Air - ONLIMO.* [Online] Available at: http://onlimo.bppt.go.id/ [Accessed 19 September 2016].

Hart, B. T., Lake, P. S., Webb, J. A. & Grace, M. R., 2003. Ecological risk to aquatic systems from salinity increases. *Australian Journal of Botany,* Volume 51, pp. 689-702.

IETF, 2009. *RFC 4919 - IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs).* [Online] Available at: https://tools.ietf.org/html/rfc4919

IETF, 2014. *RFC 7252 - The Constrained Application Protocol (CoAP).* [Online] Available at: https://tools.ietf.org/html/rfc7252 [Accessed 14 March 2017].

Kesehatan, K., 2002. *Standarisasi Kualitas Air Bersih dan Air Minum,* Jakarta: Keputusan Menteri Kesehatan RI Nomor 907/MENKES/SK/VII.

Kushalnagar, e. a., 2007. IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals. *IETF RFC - Informational,* Issue RFC 4919.

Lopez, R., 2013. *An Introduction to the Internet of Things (IoT).* San Francisco, cisco.com.

Microchip, 2008. *MRF24J40MA Data Sheet.* [Online] Available at:

http://ww1.microchip.com/downloads/en/devicedoc/70329b.pdf
[Accessed 19 July 2016].

Nurul Halimatul Asmak Ismail, R. H. K. W. M. G., 2012. A Study on Protocol Stack In 6lowpan Model. *Journal of Theoretical and Applied Information Technology,* 2(SCOPUS, ISI), p. 41.

RFC7252, 2016. *RFC 7252 - The Constrained Application Protocol (CoAP).* [Online] Available at: https://datatracker.ietf.org/doc/rfc7252/ [Accessed 28 October 2017].

Schmidt, S., 2016. *6LoWPAN: An Open IoT Networking Protocol.* San Diego, Samsung Open Source Group.

Shelby, Z., ARM, Hartke, K. & Bormann, C., 2014. *The Constrained Application Protocol (CoAP).* IETF ed. Bremen: Universitaet Bremen TZI.

Shelby, Z. & Bormann, C., 2011. *6LoWPAN: The Wireless Embedded Internet.* Wiley Series In Communications Networking & Distributed Systems ed. Torquay, UK: WILEY.

Sovani, K., 2017. *Which is the best protocol to use for IOT implementation- MQTT, CoAP, XMPP, SOAP, UPnP?.* [Online] Available at: Quora: https://www.quora.com/Which-is-the-best-protocol-to-use-for-IOTimplementation-MQTT-CoAP-XMPP-SOAP-UPnP [Accessed 28 December 2017].

Tan, J., 2014. *A Survey of Technologies in Internet of Things.* IEEE International Conference on Distributed Computing in Sensor Systems ed. Santa Clara: University Santa Clara.

Texas Instruments, T. -., 2014. *CC2538DK CC2538 Development Kit | TI.com.* [Online] Available at: http://www.ti.com/tool/CC2538DK [Accessed 21 11 2016].

Tuwanut, P., 2015. *A survey on IoT architectures, protocols, applications, security, privacy, real-world implementation and future trends.* IEEE Electronic ISBN ed. Shanghai: 11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015).

Wiguna, P., 2018. *Rancang Bangun Filter Air Berbasis Arduino Pada Penampungan Air Menggunakan Metode Fuzzy.* Malang: Fakultas Ilmu Komputer, Universitas Brawijaya.

Yi, J., 2017. *An Introduction to the Telegram Bot API.* [Online] Available at: https://towardsdatascience.com/introduction-to-the-telegram-api-b0cd220dbed2 [Accessed 28 October 2017].